MiddleWare y Applet

Guía para desarrollo de Software

consumiendo funcionalidades del MiddleWare

Rolosa HyJ S.A. - MICITT

20 de Enero de 2014

Resumen

El presente documento es una guía para desarrolladores de software útil para construir aplicaciones utilizando el Applet. Con este documento es posible crear una aplicación simple que pueda acceder a la tarjeta inteligente de Firma Digital de Costa Rica. No todos los métodos y objetos son descritos en detalle. Mayor detalle se puede encontrar en la documentación del API del Middleware.

Tabla de contenido

Pre-requisitos

Para entender los ejemplos que se proveen en este documento, es necesario conocimientos de Java 6 EE y experiencia con el desarrollo de proyectos de software utilizando Apache Maven 3.

También es necesario disponer de:

- Una Tarjeta Inteligente para Firma Digital de Costa Rica
- Un lector de tarjetas apropiado
- Certificados Digitales de la cadena de la confianza de la Firma Digital de Costa Rica
- Drivers necesarios para la Tarjeta Inteligente dependiendo el sistema operativo en el que se desee realizar los trabajos.

Los drivers y certificados digitales tienen que ser obtenidos desde el sitio de Soporte de Firma Digital de Costa Rica:

https://www.soportefirmadigital.com/sfd/default.aspx

Introducción

Es un componente para navegadores web que expone funcionalidad de la tarjeta inteligente de Firma Digital de Costa Rica a las aplicaciones web.

El Applet en la actualidad depende completamente de la librería resultante de la compilación del Middleware que sigue el estándar RSA - PKCS#11, para la interacción con las tarjetas inteligentes de la Firma Digital de Costa Rica.

El uso del Applet requiere de la estructuración de la aplicación web con distintas capas:

- La página html que lleva el Applet incrustado
- El Applet mismo, que esta contenido dentro del archivo JAR producto de la compilación del código fuente

- El Middleware que sigue el estándar RSA - PKCS#11 (El cuál ya tiene que estar disponible en la máquina cliente)

Configuración

El Applet como tal sería utilizado dentro de una página web como se muestra a continuación:

```
<script src="https://www.java.com/js/deployJava.js"></script>
<script>
    var attributes = {
        code :'be.fedict.eid.applet.Applet.class',
        archive :'eid-applet-package.jar',
        width :400,
        height :300
    };
    var parameters = {
        TargetPage :'identity-result.jsp',
        AppletService : 'applet-service',
        BackgroundColor : '#ffffff'
    };
    var version = '1.6';
    deployJava.runApplet(attributes, parameters, version);
</script>
```

La aplicación web en la cual se utiliza el Applet tiene que utilizar SSL para la seguridad en la comunicación entre el navegador web y el servidor de aplicaciones web. El Applet no continuará con su ejecución si detecta una sesión de navegador sin SSL.

El Applet tampoco continuará con su ejecución cuando detecte que no tiene privilegios suficientes para hacerlo. Esto implica que el archivo JAR del Applet debe estar firmado adecuadamente. Si el Applet no está firmado adecuadamente, este no podrá utilizar recursos locales de la máquina cliente.

Parámetros

Los parámetros de configuración disponibles para el Applet se muestran a la siguiente tabla:

Parámetro	Requerido	Descripción
TargetPage	requerido	Indica la página a la cual el Applet navegara luego de realizar la operación requerida. Ejemplo: result.jsp
AppletService	requerido	Apunta al Applet Service que reside en el componente del lado del servidor que manejara la comunicación entre el Applet y el (servlet) contenedor de la aplicación web. Ejemplo: applet-service
BackgroundColor	opcional	El color de fondo que es utilizado para la interfaz de usuario del Applet. El color por defecto es blanco. Ejemplo: #ffffff
ForegroundColor	opcional	El color externo que es utilizado para la interfaz de usuario del Applet. El color por defecto es negreo. Ejemplo: #000000
Language	opcional	El lenguaje que es utilizado para la interfaz de usuario del Applet a través de la internacionalización de los mensajes de estado. Si no es provisto, el Applet utiliza el lenguaje por defecto del JRE. Por ejemplo: es
MessageCallback	opcional	Por medio de este parámetro, un desarrollador web puede especificar un Javascript callback. La función callback sera invocada cada vez que el Applet muestre un mensaje informativo. La función se vería como: function aMessageCallback(status, message)
MessageCallbackEx	opcional	Por medio de este parámetro, un desarrollador web puede especificar un Javascript callback. La función callback sera invocada cada vez que el Applet muestre un mensaje informativo. La función se vería como: function aMessageCallback(status, messageId, message)
DiagnosticTestCallback	opcional	Por medio de este parámetro, un desarrollador web puede especificar un Javascript callback para Applets que se ejecuten en modo de diagnostico. La función callback sera invocada cada vez que el Applet realice una prueba de diagnostico

UserAgent	opcional	Por medio de este parámetro, un desarrollador web puede dejar al Applet utilizar un "User-Agent HTTP header" dado. Algunos servidores pueden requerir que el User-Agent reportado por el Applet sea el mismo que el reportado por el navegador web, para así poder utilizar el mismo contexto de sesión. Ejemplo: navigator.userAgent
HideDetailsButton	opcional	Cuando este parámetro tiene el valor: true , el Applet ocultara el botón de "detalles". Para compensar esta remoción, el Applet utilizara la Consola de Java para todos los mensajes requeridos de ser escritos.
NoChunkedTransferEncoding	opcional	Cuando este parámetro tiene el valor: true, el Applet no utilizara "chunked transfer-encoding" para la comunicación con el componente del Applet-service.

Cookies

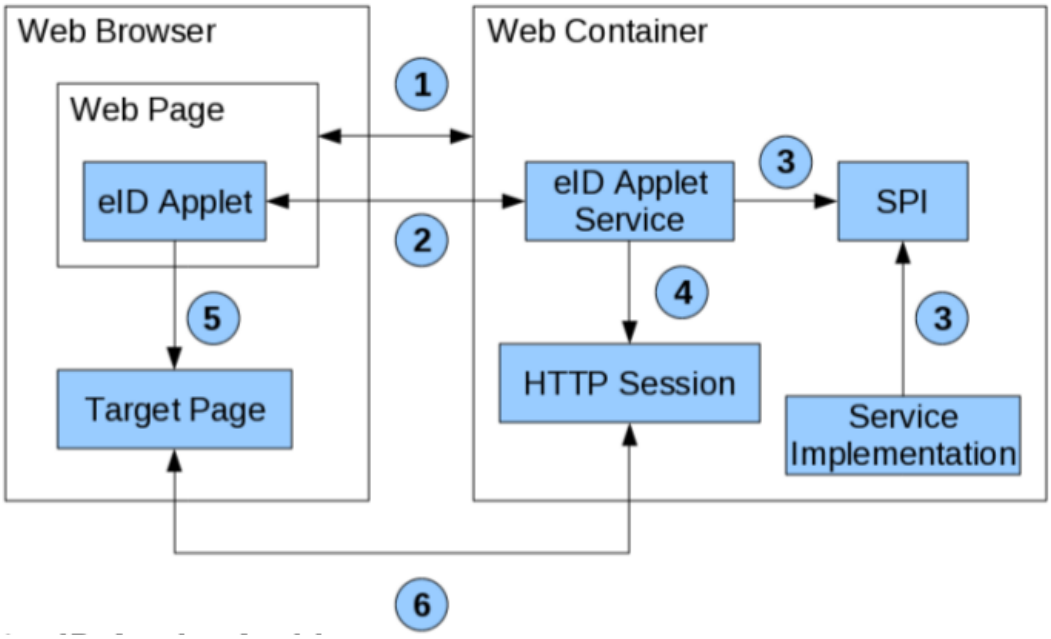
Navegadores web más recientes pueden deshabilitar cookies por defecto. Como el plugin Java para navegadores web es un plugin externo, es posible que el cookie de sesión no se comunique con el FRE. Como opción valida se puede especificar lo siguiente al final del valor para el parámetro AppletService:

;jsessionid=#{facesContext.externalContext.getSession(false.id)}

Applet Service

El Applet requiere de un componente-servicio que resida en el lado del servidor para comunicar y transferir la información de autenticación y firma desde el navegador web hacia el servidor utilizando un canal seguro. Este componente se llama **Applet Service**. El Applet SDK, viene con el **Servlet Applet Service** para facilitar la integración dentro de aplicaciones web que contengan el Applet dentro del contenedor servlet Java EE.

La arquitectura y el detalle del proceso de la información se muestran a continuación:



En el paso (1) el **Web Browser** (navegador web) carga la pagina web que contiene la referencia al Applet. El navegador web luego continua con la carga del Applet a través del plugin JRE (Java Runtime Enviroment). Una vez que el Applet ha sido completamente cargado, este mismo inicia el protocolo con el **Applet Service** (Servicio para el Applet que se soporta en el servidor) como se muestra en el paso (2). Para las operaciones requeridas como la Autenticación del Titular de algún Certificado Digital o la firma digital de documentos electrónicos, se requiere configurar los componentes necesarios en el **SPI** (Proveedor de Servicios) los cuales son llamados en el paso (3) desde el Applet Service mientras se continua con la ejecución del protocolo. Una vez terminada la ejecución del protocolo, en el paso (4) , el Applet Service envía los atributos necesarios al contexto **HTTP Session** (Sesion HTTP). Finalmente en el paso (5) el Applet hace que el navegador web se direccione a la **Target Page** (Pagina de destino), la cual puede ya acceder a los atributos de que se hicieron disponibles en el contexto HTTP Session, paso (6), y que fueron puestos a disponibilidad por el Applet Service.

El Servlet - Applet Service puede ser configurado a través del archivo web.xml (web deployment descriptor) como se muestra en el siguiente ejemplo:

```
<servlet>
    <servlet-name>AppletServiceServlet</servlet-name>
    <servlet-class>
        be.fedict.eid.applet.service.AppletServiceServlet
    </servlet-class>
</servlet>
<servlet-mapping>
    <servlet-name>AppletServiceServlet</servlet-name>
    <url-pattern>/applet-service</url-pattern>
</servlet-mapping>
```

El Applet Service, que se encuentra en el archivo JAR eid-applet-service.jar, tiene diferentes dependencias a productos de terceros. Estos productos de terceros se encuentran en el directorio lib/ dentro del paquete Applet SDK. Dependiendo del entorno de Java EE runtime con el que se disponga localmente, estos archivos JAR deberían ser colocados en el directorio META-INF/lib de la aplicación web.

En el caso de utilizar Maven como build-system, puesto que el proyecto del Appet, viene con un BOM este puede ser incluido en los archivos POM como se muestra a continuación:

```
<dependencyManagement>
    <dependencies>
        <dependency>
            <groupId>be.fedict.eid-applet</groupId>
            <artifactId>eid-applet-bom</artifactId>
            <version>1.1.3</version>
            <type>pom</type>
            <scope>import</scope>
        </dependency>
    </dependencies>
</dependencyManagement>
```

Ahora simplemente se puede agregar el Applet Service como una dependencia como se muestra a continuación:

```
<dependency>
    <groupId>be.fedict.eid-applet</groupId>
    <artifactId>eid-applet-service</artifactId>
</dependency>
```

Debido a que para se necesitan obtener los certificados digitales que se almacenan en la tarjeta inteligente, es necesario especificar el siguiente parámetro dentro del archivo de configuración del servlet:

```
<init-param>
    <param-name>IncludeCertificates</param-name>
    <param-value>true</param-value>
</init-param>
```

Autenticación

El Applet puede ser utilizado para autenticar a un usuario a través de la tarjeta inteligente.

Para realizar la autenticación, es necesario implementar la interfaz **AuthenticationService**. Este interfaz es parte del proyecto **eid-applet-service-spi**. Este componente-servicio (EJB3) session-bean tiene que ser registrado en algún lugar en JNDI y la ubicación JNDI tiene que ser informada a través del **init-param** al **AppletServiceServlet** como se muestra a continuación:

```
<init-param>
    <param-name>AuthenticationService</param-name>
    <param-value>your/location/in/jndi/AuthenticationServiceBean</param-value>
</init-param>
```

Por defecto el Applet realizara una firma digital una secuencia de datos (salt, challenge) utilizando la clave privada de autenticación que se encuentra en la tarjeta inteligente.

La información formada es enviada sobre SSL por el AppletService. El valor del dato **salt** es producido por el mismo Applet.

Firmas

El Applet también puede ser utilizado para la creación de firmas digitales utilizando el certificado de Firma Digital/No-repudio que se encuentra dentro de la tarjeta inteligente. El algoritmo soportado para la firma actualmente es: "SHA1-RSA-PKCS1".

Para utilizar esta funcionalidad es necesario implementar la interfaz **SignatureService**. . Este interfaz es parte del proyecto **eid-applet-service-spi**. Este componente-servicio (EJB3) session-bean tiene que ser registrado en algún lugar en JNDI y la ubicación JNDI tiene que ser informada a través del **init-param** al **AppletServiceServlet** como se muestra a continuación:

```
<init-param>
    <param-name>SignatureService</param-name>
    <param-value>your/location/in/jndi/SignatureServiceBean</param-value>
</init-param>
```

Durante la fase de pre-firma, se puede recibir el certificado de no-repudio configurando lo siguiente:

```
<init-param>
    <param-name>IncludeCertificates</param-name>
    <param-value>true</param-value>
</init-param>
```

El tipo de firma digital creado por el Applet es completamente determinado por la implementación del SignatureService SPI. Se proveen diversas implementaciones base del SignatureService SPI como parte del proyecto eid-applet-service-signer.

Las implementaciones de servicio de firma más importantes provistas por el Applet SDK son:

- Firmas ODF 1.2 (Soportadas por OpenOffice.org 3.1/3.2)
- Office OpenXML 9Soportadas por Microsoft Office 2007/2010
- Firmas CMS (PKCS#7)

Auditoria

Para poder obtener una traza de las actividades realizadas en el Applet Service por los clientes, el Applet Service ofrece el soporte para auditoria.

Para utilizar esta funcionalidad es necesario implementar la interfaz **AuditService**. . Este interfaz es parte del proyecto **eid-applet-service-spi**. Este componente-servicio (EJB3) session-bean tiene que ser registrado en algún lugar en JNDI y la ubicación JNDI tiene que ser informada a través del **init-param** al **AppletServiceServlet** como se muestra a continuación:

```
<init-param>
    <param-name>AuditService</param-name>
    <param-value>your/location/in/jndi/AuditServiceBean</param-value>
</init-param>
```

Integración con Apache Maven

En esta sección se mostrara lo relacionado a la integración del Applet dentro de proyectos basados en Maven.

Primeramente, se debe agregar el repositorio de e-contract al archivo de proyecto pom.xml como se muestra a continuación:

```
<repository>
    <id>e-contract</id>
    <url>https://www.e-contract.be/maven2/</url>
    <releases>
        <enabled>true</enabled>
    </releases>
</repository>
```

Esto es necesario debido a que el Applet requiere de muchas dependencias producto de desarrollo de terceros y estas dependencias están disponibles en el repositorio de e-contract.

La integración del Applet dentro de una aplicación web Java EE consiste de 2 pasos. Primero se debe agregar el Applet como un recurso web a la aplicación web, para esto agregar las siguientes dependencias dentro del bloque **<dependencies>:**

```
<dependency>
    <groupId>be.fedict.eid-applet</groupId>
    <artifactId>eid-applet-package</artifactId>
    <scope>provided</scope>
</dependency>
<dependency>
    <groupId>be.fedict.eid-applet</groupId>
    <artifactId>eid-applet-js</artifactId>
    <scope>provided</scope>
</dependency>
```

Luego ya se puede incluir el archivo JAR del Applet y su correspondiente Javascript como recurso web como se muestra a continuación:

```
<plugin>
    <groupId>org.apache.maven.plugins</groupId>
    <artifactId>maven-dependency-plugin</artifactId>
    <executions>
        <execution>
            <id>unpack</id>
            <phase>process-resources</phase>
            <goals>
                <goal>unpack</goal>
            </goals>
            <configuration>
                <artifactItems>
                    <artifactItem>
                        <groupId>be.fedict.eid-applet</groupId>
                        <artifactId>eid-applet-js</artifactId>
                        <outputDirectory>
                              ${project.build.directory}/${project.artifactId}-
${project.version}
                        </outputDirectory>
                    </artifactItem>
                </artifactItems>
                <includes>*.js</includes>
            </configuration>
        </execution>
        <execution>
            <id>copy</id>
            <phase>process-resources</phase>
            <goals>
                <goal>copy</goal>
            </goals>
            <configuration>
                <artifactItems>
                    <artifactItem>
                        <groupId>be.fedict.eid-applet</groupId>
                        <artifactId>eid-applet-package</artifactId>
                        <type>jar</type>
                        <outputDirectory>
                              ${project.build.directory}/${project.artifactId}-
${project.version}
                        </outputDirectory>
                    </artifactItem>
                </artifactItems>
            </configuration>
        </execution>
    </executions>
</plugin>
```

La inclusión del Applet Service depende principalmente de la funcionalidad que se desee utilizar, sin embargo, la mayoría del tiempo la siguiente configuración de dependencia debería ser suficiente:

```
<dependency>
    <groupId>be.fedict.eid-applet</groupId>
    <artifactId>eid-applet-service</artifactId>
    <exclusions>
        <exclusion>
            <groupId>javax.servlet</groupId>
            <artifactId>servlet-api</artifactId>
        </exclusion>
        <exclusion>
            <groupId>com.lowagie</groupId>
            <artifactId>itext</artifactId>
        </exclusion>
        <exclusion>
            <groupId>com.googlecode.json-simple</groupId>
            <artifactId>json-simple</artifactId>
        </exclusion>
    </exclusions>
</dependency>
```

Applet SDK

Este proyecto permite la generación del SDK correspondiente a la versión en la que se encuentra el proyecto principal. Realiza una compilación de los proyectos anteriormente listados y recolecta los archivos JAR resultantes dentro de un solo paquete. Para este proceso es necesario utilizar un eToken que contenga el certificado digital de firma para firmar el archivo JAR del Applet.

Para realizar la generación del paquete SDK se debe utilizar el siguiente comando desde una ventana de consola (suponemos la existencia de la instalación de Apache Maven 3 en el sistema sin incluir el soporte nativo de Maven que NetBeans 6.9.1 presenta):

mvn -Dhttp.proxyHost=proxy.suservidor.net -Dhttp.proxyPort=8080 -Psdk,etoken clean deploy

El SDK resultante será ubicado en:

eid-applet-sdk/target/

Aplicación demostrativa - Java 6 EE

El código fuente y proyecto de Apache Maven soportado por NetBeans 6.9.1 de la aplicación demostrativa en Java **JAVAmwEIDTest** puede ser obtenido desde el sitio SVN de Google en la ruta:

trunk\aplicaciones_demostrativas\WEB\MavenEnterpriseApp

El detalle de utilización de la aplicación demostrativa web puede ser encontrado en el documento denominado:

5.1 Manual JAVA6EE Applet Test.pdf