


# ESCUELA TÉCNICA SUPERIOR DE INGENIEROS INDUSTRIALES Y DE TELECOMUNICACIÓN

Titulación:

INGENIERO TÉCNICO EN INFORMÁTICA DE GESTIÓN

Título del proyecto:

## SISTEMA DE GESTIÓN DE INFORMACIÓN EPIDEMIOLÓGICA EN EMERGENCIAS INTERNACIONALES

## MEMORIA

Daniel Mayor Azcona

Jon Legarrea Oteiza

Alfredo Pina Calafi

Pamplona, 23 de marzo de 2010

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

Este Proyecto va especialmente dedicado a mi infinitamente paciente familia, a mi compañero de fatigas Eder, a Jon por mostrarme su apoyo y animarme en los momentos difíciles, a Alfredo por codirigirme el Proyecto desde países remotos, al experimentado Eneko por arrojarme luz desinteresadamente y a Raúl por ser el padrino e inspirador del Proyecto



upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

# Índice





upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

# Introducción

Como objeto principal se plantea el desarrollo de sistemas de gestión de información en tiempo real de la información operativa para la toma de decisiones en ámbitos de emergencia de ámbito local, regional e internacional. Para ello nos planteamos el estudio de sistemas compatibles con diferentes plataformas a nivel de aplicación móvil y de servidor.

A nivel de servidor es preferible disponer de plataformas que puedan ser implementados en la mayoría de sistemas operativos pensados para servidores, principalmente en plataformas de software libre GNU/Linux como en propietarias Windows/Mac.

En cuanto a los dispositivos móviles, es importante su simplicidad de manejo, que debe ser rápido, intuitivo y con escaso consumo de recursos hardware.

Por último, es importante que la tecnología inalámbrica para los dispositivos móviles sea flexible, posibilitando usar la que esté disponible en el momento. El sistema desarrollado debe tener en cuenta que será operado por técnicos de intervención en emergencias y especialistas de coordinación operativa en situaciones de estrés, por lo que debe ser un complemento al trabajo que realizan y servir como herramienta. Es decir, debe consumir poco esfuerzo y tiempo para ser operado.

En todo caso, se trata de abrir líneas de trabajo en este tipo de tecnologías y problemáticas de manera coordinada e integrada con los actores implicados. Para demostrar la viabilidad de lo expuesto anteriormente se desarrollará una aplicación concreta que cumpla los requisitos planteados.

Como ejemplo claro y real de aplicaciones operativas de gestión de información en entornos de emergencias, se ha seleccionado la gestión de información epidemiológica en catástrofes internacionales. Esta gestión implica implica una red de apoyo compuesta por personal civil: grupos de búsqueda y rescate, médicos, logistas, técnicos de campo y demás especialistas.

Se requiere un sistema gráfico que permita introducir un informe diario sobre los pacientes atendidos por el personal sanitario desplazado, clasificados por su sintomatología, rango de edad y sexo. Una de las



upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

principales demandas del sistema es el almacenamiento de información en el propio dispositivo móvil.

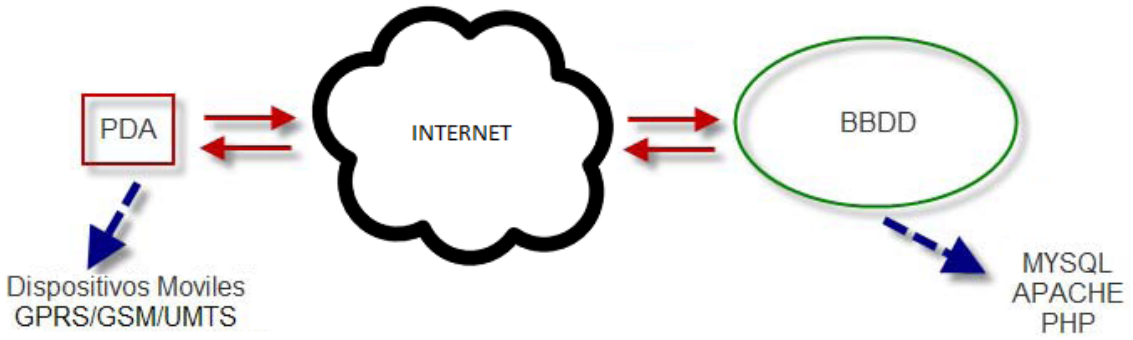
## *Antecedentes*

La gestión de la información en entornos de emergencia es una herramienta clave en los procesos de tomas de decisión. Los sistemas existentes son excesivamente caros en su implantación y funcionamiento y no es viable implantarlos para la asistencia a víctimas en los países en vías de desarrollo. A esto se suma tanto el costo de licencias como el de los aparatos y el de formación del personal.

Esta línea de proyectos ha estudiado las tecnologías existentes y ha determinado un modelo tecnológico de reducido costo y gran capacidad de adaptación a diferentes sistemas operativos y plataformas de conectividad para la transmisión de información en entornos de emergencia. Como ejemplo de aplicación se ha desarrollado un programa para la gestión de información epidemiológica en emergencias internacionales. Este caso es propio de la acción internacional, por lo que contiene todas las limitaciones de conectividad de los países en vías de desarrollo y unos requerimientos mayores de actualización de datos. Para responder a estas necesidades hemos decidido:

- Que la aplicación sea multiplataforma:
  - A nivel de servidor.
  - A nivel de aplicación de usuario.
- Que consuma el menor ancho de banda posible.
- Que utilice, en la medida de lo posible, software libre.

Con estas premisas se ha puesto en marcha un servidor de base datos MySQL con un servidor web Apache y como lenguaje activo PHP.
A nivel de cliente se han desarrollado aplicaciones en Java ME, las que mediante POST con el PHP escriben en la base de datos.






upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

Este caso es propio de la acción internacional, por lo que contiene todas las limitaciones de conectividad de los países en vías de desarrollo y unos requerimientos mayores de frecuencias de refresco y actualización de datos.

## *Datos de partida*

Dentro de la acción humanitaria, cada vez se avanza más en la gestión de la información en grandes desastres tanto naturales como antrópicos. Esta gestión se hace más necesaria en aras de una mejor coordinación en terreno, con las sedes centrales de las organizaciones de respuesta y socorro así como las autoridades pertinentes.

Han sido varias las iniciativas de implantación de sistemas de gestión que se han visto frenadas por tratarse de aplicaciones pesadas que requieren conexiones de banda ancha.

Este proyecto ha trabajado este extremo para que el volumen sea mínimo. El proyecto abre la puerta para sistemas de información en tiempo real independientemente del sistema de conectividad, sea RTB, redes GSM/UMTS o satelital Iridum, Thuraya o Inmarsat a costos asumibles.

La comunicación de datos en tiempo real puede ser de gran utilidad para sistemas logísticos, de control epidemiológico y en los procesos de evaluación de impacto. En muchos casos la respuesta tarda en darse por la dificultad de evaluación. La correcta respuesta ante desastres, si bien no es un motor de desarrollo humano sí que ayuda a reducir el impacto de estos en los procesos abiertos. De esta forma, pese a que este proyecto no es una aportación directa al desarrollo social, sí que puede ayudar a mitigar su afección por los desastres y la posibilidad de ser retomados.

Los datos de partida han sido proporcionados por Cruz Roja, con hojas de cálculo provenientes la operación de 2008 en Gonaïves, Haití con datos epidemiológicos reales recolectados in situ.

Los requerimientos de la aplicación son:

Aplicación móvil que abarque un amplio abanico de modelos de teléfonos móviles y PDAs, sin olvidarse de la gama baja.

La aplicación será utilizada por técnicos sanitarios que introducirán un resumen diario con el número de personas atendidas y los síntomas que



upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

presentaban clasificados por sexo y grupo de edad (menores y mayores de 5 años).

Almacenamiento de información offline porque no hay conectividad permanente.

Posibilidad de conectividad directa a la base de datos central si hay cobertura de datos, o alternativamente poder transmitirlos a una base intermedia por Wi-Fi, cable de datos o tarjeta de memoria, que es la que realizará el envío.



upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

# Análisis de tecnologías

## *Plataformas móviles*

Cuando se enfocó el proyecto, sirvió de inspiración la experiencia del trabajo anterior de Mikel Belzunegi y Rafa Huarte, dentro de esta línea de proyectos, *Estudio de la problemática del tratamiento de datos y de su visualización en situaciones de emergencia e implementación de una solución basada en PDA, PC y servidor*, donde habían desarrollado una aplicación móvil con Adobe Flash Lite.

Emplearon esta plataforma por diferentes razones. Por una parte les permitía desarrollar una aplicación con un entorno gráfico interactivo y amigable sin necesidad de ser expertos en diseño. Estas aplicaciones compiladas, pese a ser eminentemente gráficas, requieren poco espacio de almacenamiento debido a su naturaleza vectorial, lo que reduce los requerimientos técnicos de los terminales móviles. Además, Flash se considera multiplataforma, ya que dispone tanto de versiones para ordenadores de sobremesa como para dispositivos móviles.

Actualmente, el panorama de aplicaciones para dispositivos móviles está viviendo una revolución, siendo ésta liderada por las plataformas iPhone de Apple y Android de Google, que compiten ferozmente por hacerse con el mercado. Windows Mobile y BlackBerry intentan mantener su nicho, además de Symbian de Nokia y Palm webOS. Precisamente Symbian y Windows Mobile, que hasta ahora eran los más extendidos, son los únicos que soportan Flash Lite.

Aunque el futuro se muestra prometedor, el inconveniente de desarrollar aplicaciones específicas para dichas plataformas es que son incompatibles entre sí, además de que hasta ahora están vinculadas a dispositivos móviles de gama media-alta, y en este proyecto se pretendía llegar al mayor número posible de móviles, incluyendo los de gama baja. Por tanto, las alternativas que nos quedaban eran la plataforma Java Micro Edition (J2ME) de Sun Microsystems y Adobe Flash Lite. Obviamente, cada una de ellas tiene sus ventajas e inconvenientes que, resumidos, son los siguientes:

### Adobe Flash Lite

Esta plataforma emplea ActionScript, un lenguaje de programación de script. Es decir, no requiere la creación de un programa completo para que la aplicación alcance los objetivos. Este lenguaje está basado en



upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

especificaciones de estándar de industria ECMA-262, un estándar para Javascript, de ahí que ActionScript se parezca tanto a JavaScript.

Por tanto, se basa en programar scripts asociados a fotogramas independientes, por lo que resulta algo problemático a la hora de estructurar la ejecución. La forma de trabajar con fotogramas es una característica clave de Flash, y esto conlleva ciertas particularidades, por lo que la codificación del proyecto resultó muy compleja para que el comportamiento del programa fuera el de una aplicación convencional.

Además, aunque la versión ActionScript 3 se ha convertido en un lenguaje orientado a objetos como lo es Java, al ser un lenguaje orientado a web tiene menos versatilidad. Si bien cuenta con una API que va mejorando progresivamente, lleva varios años en desventaja con la de Java. Contar con una API tan básica limita mucho el número de funciones disponibles para tareas con cierta complejidad, y habría que estar implementando código que en otros lenguajes ya está solucionado.

Así que, después de estudiar las limitaciones que nos imponía Flash a la hora de programar el sistema, nos motivó a pensar en la alternativa de Java. Fue de vital importancia la cuestión de que la mayoría de las aplicaciones para dispositivos móviles que se desarrollan en la actualidad son en este lenguaje.

### Java Micro Edition

Java es un lenguaje de programación orientado a objetos desarrollado por Sun Microsystems (actualmente propiedad de Oracle) a principios de los años 90. El lenguaje en sí mismo toma mucha de su sintaxis de C y C++, pero tiene un modelo de objetos más simple y elimina herramientas de bajo nivel, que suelen inducir a muchos errores, como la manipulación directa de punteros o memoria.

Que Java sea un lenguaje orientado a objetos ("OO") se refiere a un método de programación y al diseño del lenguaje. Aunque hay muchas interpretaciones para OO, una primera idea es diseñar el software de forma que los distintos tipos de datos que usen estén unidos a sus operaciones. Así, los datos y el código (funciones o métodos) se combinan en entidades llamadas objetos. Un objeto puede verse como un paquete que contiene el "comportamiento" (el código) y el "estado" (datos). El principio es separar aquello que cambia de las cosas que permanecen inalterables.



upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

Frecuentemente, cambiar una estructura de datos implica un cambio en el código que opera sobre los mismos, o viceversa. Esta separación en objetos coherentes e independientes ofrece una base más estable para el diseño de un sistema software. El objetivo es hacer que grandes proyectos sean fáciles de gestionar y manejar, mejorando como consecuencia su calidad y reduciendo el número de proyectos fallidos. Otra de las grandes promesas de la programación orientada a objetos es la creación de entidades más genéricas (objetos) que permitan la reutilización del software entre proyectos, una de las premisas fundamentales de la Ingeniería del Software. Un objeto genérico "cliente", por ejemplo, debería en teoría tener el mismo conjunto de comportamiento en diferentes proyectos, sobre todo cuando estos coinciden en cierta medida, algo que suele suceder en las grandes organizaciones. En este sentido, los objetos podrían verse como piezas reutilizables que pueden emplearse en múltiples proyectos distintos, posibilitando así a la industria del software a construir proyectos de envergadura empleando componentes ya existentes y de comprobada calidad; conduciendo esto finalmente a una reducción drástica del tiempo de desarrollo. Podemos usar como ejemplo de objeto el aluminio. Una vez definidos datos (peso, maleabilidad, etc.), y su "comportamiento" (soldar dos piezas, etc.), el objeto "aluminio" puede ser reutilizado en el campo de la construcción, del automóvil, de la aviación, etc.

La reutilización del software ha experimentado resultados dispares, encontrando dos dificultades principales: el diseño de objetos realmente genéricos es pobremente comprendido, y falta una metodología para la amplia comunicación de oportunidades de reutilización. Algunas comunidades de "código abierto" quieren ayudar en este problema dando medios a los desarrolladores para diseminar la información sobre el uso y versatilidad de objetos reutilizables y bibliotecas de objetos.

Las aplicaciones Java están típicamente compiladas en un bytecode, aunque la compilación en código máquina nativo también es posible. En el tiempo de ejecución, el bytecode es normalmente interpretado o compilado a código nativo para la ejecución, aunque la ejecución directa por hardware del bytecode por un procesador Java también es posible.

Java está muy extendido en terminales móviles y goza de gran aceptación. Además, al ser multiplataforma se puede reutilizar el código para diferentes plataformas. Tiene una API muy completa, extensa y madura, lo que conlleva gran facilidad para programar aplicaciones robustas y estables. Además, existen entornos de desarrollo integrados como

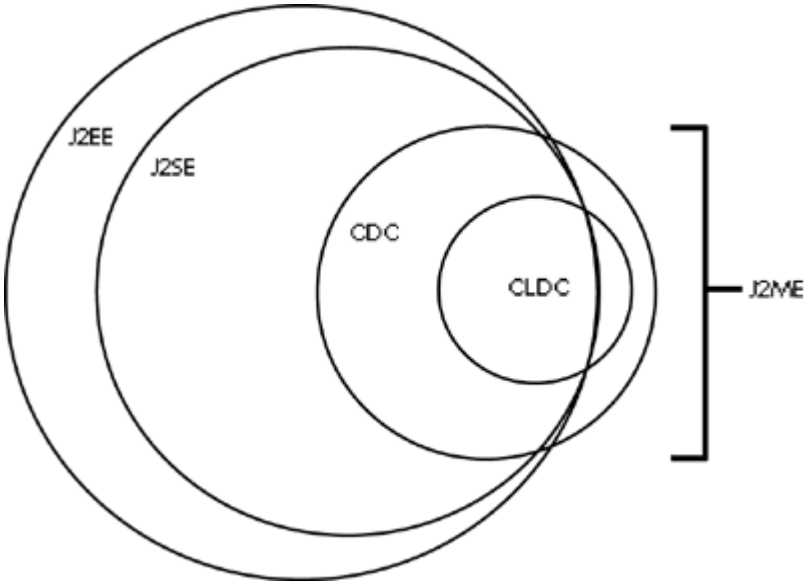


upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

NetBeans, Eclipse, o JBuilder, que facilitan tareas de cierta complejidad, tales como generar interfaces gráficas con ventanas y botones.

Proporciona una manera sencilla de documentar el código usando Javadoc, que es una utilidad que genera documentación de APIs en formato HTML a partir de código fuente Java. Actualmente, Javadoc es el estándar de la industria para documentar clases de Java. La mayoría de IDEs lo generan automáticamente.

En términos generales, se puede decir que este lenguaje es el más extendido del sector. El número de aplicaciones desarrolladas en Java Micro Edition (también conocido como Java ME o J2ME) es enorme, el lenguaje cuenta con el respaldo incondicional de Sun, y se ha convertido en el estándar de facto en el desarrollo de aplicaciones para móviles. Básicamente, es un subconjunto de Java Platform, Enterprise Edition (Java EE o J2EE), por lo que casi cualquier desarrollador de Java puede ponerse a programar para móviles tras un breve periodo de adaptación al medio. Además, Sun Microsystems liberó la mayor parte de Java bajo la licencia GNU GPL, por lo que no hay que pagar por su uso.




Por tanto, el hecho de disponer de una API tan completa, ser uno de los lenguajes más extendidos, contar con cierta experiencia previa en programación en esta plataforma, así como una disponibilidad multiplataforma similar a Flash, nos hizo decantarnos por Java.



upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

Sin embargo, como sucede en general con todas las tecnologías basadas en Java, tiene sus inconvenientes. Es complicado escribir un programa en Java Micro Edition con unas garantías mínimas de que vaya a poder funcionar en un amplio abanico de dispositivos, sobre todo cuando se intenta utilizar alguna de las APIs que permiten acceder al hardware del teléfono, como puede ser el Bluetooth o el GPS.

## *Almacenamiento de la información*

Respecto al almacenamiento de información, necesitábamos un sistema que almacenara un registro diario de información sin apenas complejidad, pero cuyo requisito imprescindible era poder trabajar en local en condiciones de conectividad nula.

### SQLite

Una de las posibilidades que se barajó fue SQLite. Se trata de una biblioteca de software que implementa un motor de bases de datos SQL transaccional, contenida en sí misma, que no necesita un servidor ni ser configurada. SQLite es, probablemente, el motor de bases de datos SQL más extendido en la actualidad. Además, su código fuente es de dominio público.

Para manejar eficientemente los motores de los diferentes gestores de bases de datos se proporcionan unas APIs con bibliotecas de métodos escritas en Java, llamadas JDBC (Java Database Conectivity).

Para comenzar con la resolución del proyecto se decidió hacer una biblioteca en Java para interactuar con SQLite que contuviera funciones o métodos sencillos con operaciones básicas sobre una base de datos, que pudieran ser utilizados continuamente durante el proyecto. Como Java ME no dispone de clases para manejar SQL, decidimos hacer la biblioteca en Java SE ya que contábamos con experiencia previa en esta plataforma. Más adelante empezaríamos a desarrollar en Java ME.

Los métodos son:

- Conectar: conexión a la base de datos
- Desconectar: desconexión de la base de datos
- Escribir_1: escribe un atributo de un registro
- Leer_1: lee un atributo de un registro



upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

- Escribir: escribe un nuevo registro completo
- Leer: lee un nuevo registro completo

### Dato

Para facilitar el manejo de datos con esta biblioteca se creó un nuevo objeto llamado Dato. La estructura de este objeto contiene tres elementos:

- Id: identificador del registro de la base de datos
- Nombre: nombre de un atributo de la base de datos
- Valor: valor de dicho atributo.

Se trata de un objeto simple con *getters* y *setters* para acceder a los atributos de dicho objeto y métodos *equal* y *toString* para poder compararlos entre sí y poder imprimirlos con un formato específico, además del constructor.

Dicha biblioteca se diseñó para que tuviera la flexibilidad de ser capaz de manejar diferentes formatos de atributo y tablas de bases de datos. De esta manera era posible crear un objeto que fuera "totalmente" genérico y sirviera tanto para bases de datos SQLite como MySQL.

### BaseDeDatos

Es la clase que contiene los métodos para manejar los datos. Tiene unos atributos que le permiten conectarse a una base de datos:

- Nombre: nombre de la base de datos.
- Dirección: URL o IP del servidor de la base de datos. SQLite no lo implementa ya que trabaja en local.
- Puerto: puerto de conexión del servidor de la base de datos.
- Driver: driver necesario para la conexión a la base de datos. Con valor 0 utiliza SQLite, con 1 utiliza MySQL, con 2 utiliza Oracle y con 3 utiliza PostgreSQL. En esta versión la biblioteca sólo está implementada para SQLite y MySQL local.
- Usuario: cuenta de usuario para conectarse a la base de datos. SQLite no lo emplea.
- Contraseña: contraseña para conectarse a la base de datos. SQLite no lo emplea.

Como se puede observar, hay varios atributos de los que SQLite prescinde, como son dirección, usuario y contraseña. Para resolver este problema se crean dos constructores. Uno específico para SQLite y otro preparado para el resto de SGBD (en la versión desarrollada tan sólo para MySQL). Una



upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

vez de creado el objeto, ya se pueden comenzar a utilizar los métodos nombrados en la biblioteca.

### Conexión a la base de datos

La conexión a un sistema de base de datos u otro es difiere debido a las diferentes tecnologías de cada uno, pero el método *getConnection* de la clase *DriverManager* varía de SQLite al resto de SGBD. Se ha dejado preparado para los SGBD como Oracle y PostgreSQL, aunque no están habilitados puesto que no se han probado. Se pueden implementar más SGBD pero habría que añadir las bibliotecas para poder manejarlos en Java, además de añadirle al método el código necesario para que devuelva un objeto *Connection*. Dicho método devuelve un objeto del tipo *Connection*, que es el que se emplea para conectarse a la base de datos. El código es el siguiente:

```
/**
* Este método realiza la conexión a la base de datos indicada por getNombre()
* @exception ClassNotFoundException devuelve ClassNotFoundException si no existe la clase
* @exception SQLException devuelve SQLException si no ha podido conectarse a dicha base de datos.
* @return devolver Devuelve un objeto tipo conexion con esa base de datos
*/

public Connection creaConexion() throws SQLException, ClassNotFoundException{
        Connection devolver = null;
        String nameBD = getNombre();
        int draiver = getDriver();
        if (draiver == 0){
                //SQLite
```



upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

```
            Class.forName("org.sqlite.JDBC");
            devolver =
        DriverManager.getConnection("jdbc:sqlite:"+nameBD);
        }
        else if(draiver == 1){
            //MySQL
            Class.forName("org.gjt.mm.mysql.Driver");
            String direc = getDireccion();
            String puert = getPuerto();
            String usuari = getUsuario();
            String cntraseña = getContraseña();
            devolver = DriverManager.getConnection("jdbc:mysql://" +
        direc + ":" + puert + "/" + nameBD, usuari , cntraseña);
        }
        // else if (draiver == 2){
        // //Oracle
        // Class.forName("oracle.jdbc.driver.OracleDriver");
        // String direc = getDireccion();
        // String puert = getPuerto();
        // String usuari = getUsuario();
        // String cntraseña = getContraseña();
        // devolver = DriverManager.getConnection("jdbc:oracle:thin:@"+
    direc + ":" + puert + ":" + nameBD, usuari, cntraseña);
        // }
        // else if (draiver == 3){
            // //PostgreSQL
            // Class.forName("org.postgresql.Driver");
```



upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

```
            // String direc = getDireccion();
            // String puert = getPuerto();
            // String usuari = getUsuario();
            // String cntraseña = getContraseña();
            // devolver =
        DriverManager.getConnection("jdbc:postgresql://" + direc + ":" +
        puert + "/" + nameBD, usuari, cntraseña);
        // }
        else{
            //No se ha puesto esta opcion asi que fuera de aki
            System.out.println("Este driver no está implementado:
        "+draiver+".");
        System.exit(0);
        }
        return devolver;
}
```

Para poder enviar consultas SQL a la base de datos se emplea la clase Statement. Ésta se obtiene a través del objeto *Connection* especificado previamente. La operación es efectuada a través del siguiente método:

```
/**
* Este método crea el objeto Statement en el que se ejecutan las consultas.
* @param con Conexión a la Base de datos.
* @exception SQLException devuelve SQLException si por alguna causa
no se ha podido crear el objeto
* @return devolver Devuelve un objeto Statement
*/
```



upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

```
public Statement daStatement (final Connection con)throws
SQLException{
        Statement devolver = con.createStatement();
        return devolver;
}
```

### Desconexión de la base de datos

Para desconectarse de una base de datos en Java hay que cerrar los objetos de la clase *Statement* y *Connection*. A este método le pasamos ambos objetos por parámetros.

```
/**
* Sirve para desconectarse de la base de datos actual.
* @param conn Conexión a la base de datos
* @param sta Objeto Statment para realizar consultas
* @exception SQLException devuelve SQLException si no ha podido
desconectarse a dicha base de datos
*/
public void desconectarse (Connection conn, Statement sta) throws
SQLException{
        sta.close();
        conn.close();
}
```

### Escribir_1

El siguiente método de la biblioteca es escribir_1. Este método en un principio un principio debe meter un valor. Tiene varios parámetros como tab, que es el nombre de la tabla de la base de datos donde se quiere escribir el dato, var el nombre de la variable donde se quiere escribir, val el valor de la variable a escribir, ident el identificador donde se inserta el registro y finalmente sta que es el objeto de clase Statement donde se



upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

ejecutaran las sentencias SQL. Cabe destacar que si el nombre de la variable es id (el identificador) hay que crear un nuevo registro que contendrá el resto con valores null. Si no actualizara el registro.

```
/**
* Escribe un valor en la tabla y variable que se indique de la Base de datos
* @param tab Nombre de la tabla de la base de datos donde queremos guardar el dato
* @param var Nombre de la variable que se quiere guardar el dato
* @param val Valor de la variable de la que se quiere guardar el dato
* @param ident Identificador del registro donde se inserta el dato
* @param sta Objeto Statement donde se ejecutan las inserciones o actualización.
*/
public void escribir_1(final String tab, final String var, final String val, final int ident, final Statement sta){
        //Se desecha la opcion de conectar y desconectar cada vez que se va a lanzar este método
        try{
                String insercion;
                if (var.equals("id")){
                        insercion = "INSERT INTO "+tab+" ("+var+") VALUES ('"+val+"');";
                }
                else{
                        insercion = "UPDATE "+tab+" SET "+var+" = '"+val+"' WHERE id = "+ident+";";
                }
                sta.executeUpdate(insercion);
```



upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

```
        }
        catch (SQLException sqle){
                sqle.printStackTrace();
        }
}
```

**Escribir:**

Este método simplemente se conecta a la base de datos, escribe en la tabla que se le ha pasado por parámetro el vector de Datos que se le ha pasado de la misma manera, y al finalizar la escritura de todo los datos del vector se desconecta.

```
/**
* Método que inserta un Vector cuyos elementos contienen un nombre de variable, su valor, y el id del registro
* @param tab Nombre de la tabla donde se quiere hacer la inserción de datos
* @param datos Vector que contiene todos los datos que queremos insertar
*/

public void escribir (final String tab, final Vector datos){
        //declaramos las variables para conectarse
        Connection con = null;
        Statement sta = null;
        try{
                con = creaConexion();
                sta = daStatement(con);
                Dato var = null;
                for (int i = 0; i<datos.size(); i++){
```



upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

```
                    try{
                        var = (Dato) datos.elementAt(i);
                        escribir_1(tab, var.getNombre(), var.getValor(),
                    var.getId(), sta);
                    }
                    catch (NoSuchElementException nsee){
                        System.out.println("El ultimo elemento del vector
                    no tenia valor");
                    }
                }
            desconectarse(con, sta);
            }
            catch (SQLException sqle){
                sqle.printStackTrace();
            }
            catch (ClassNotFoundException cnfe){
                cnfe.printStackTrace();
            }
}
```

**Leer_1**

El modo es muy semejante al anterior, *escribir_1*. Recibe por parámetros el nombre de la base de datos donde se encuentra la variable que se quiere leer, el nombre de la variable y su identificador. Ahora se necesita una nueva clase llamada ResultSet que representa una conexión con los datos. El método *executeQuery()* del objeto *Statement* devolverá un objeto de tipo *ResultSet*. Al obtener los resultados muestra el String de la variable var y se cierra el *ResultSet*, para después devolver el resultado obtenido.

```
/**
```



upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

```
* Método que lee de la base de datos un dato. Devuelve null si no ha encontrado nada.

* @param tab Tabla donde se encuentra el dato a leer

* @param var Nombre de la variable de la que se desea leer su valor

* @param ident Identificador del registro donde se encuentra el valor a leer

* @param sta Objeto Statement en el que se hace la consulta.

* @return String que contiene el valor del dato

*/

public String leer_1 (final String tab, final String var, final int ident, final Statement sta){

	String devuelve = null;

	String consulta = "SELECT "+var+" FROM "+tab+" WHERE id = "+ident+";";

	System.out.println("La consulta es "+consulta);

	try{

		ResultSet rs = sta.executeQuery(consulta);

		//System.out.println("El nombre del atributo que queremos saber es "+var);

		rs.next();

		devuelve = rs.getString(var);

		rs.close();

	}

	catch(SQLException sqle){

		sqle.printStackTrace();

	}

	finally{

		return devuelve;
```



upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

```
        }
}
```

### Leer

Este método es casi idéntico al anterior. Pero, a diferencia del método Escribir, que llamaba a Escribir_1 tantas veces como datos tuviese el vector de datos, éste lee todo de una sola vez debido a que se lo permite la propia naturaleza de la operación de lectura. Si se procediera a leer de uno en uno se consumirían muchos recursos inútilmente abriendo y cerrando objetos *ResultSet*. De *ResultSet* se obtiene un *ResultSetMetaData* que contiene todos los valores que hay con el identificador "reg". Después se añade cada dato en un Vector que será devuelto.

```
/** Metodo que lee un registro entero de la base de datos y lo mete en el vector. Devuelve null si no ha encontrado nada.
* @param tab Tabla donde se encuentra el registro a leer
* @param reg Numero de registro
* @param sta Objeto Statement en el que se hace la consulta
* @return Vector que contiene el registro entero
*/
public Vector leer (final String tab, final int reg, final Statement sta){
        Vector devuelve = new Vector();
        String consulta = "SELECT * FROM "+tab+" WHERE id="+reg+";";
        try{
                ResultSet rs = sta.executeQuery(consulta);
                ResultSetMetaData metaDatos = rs.getMetaData();
                int numeroColumnas = metaDatos.getColumnCount();
                //System.out.println("numero de columnas "+numeroColumnas);
                Dato datito;
```



upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

```
                String name;
                String valor;
                rs.next();
                for (int i = 0; i < numeroColumnas; i++){
                        name = metaDatos.getColumnLabel(i + 1);
                        valor = rs.getString(name);
                        datito = new Dato (name, valor, reg);
                        devuelve.addElement(datito);
                }
                rs.close();
        }
        catch(SQLException sqle){
                sqle.printStackTrace();
        }
        finally{
        return devuelve;
        }
}
```

La biblioteca funciona sin problemas con bases de datos SQLite, que era la tecnología que en principio iba a emplearse en el proyecto. En cambio, con bases de datos MySQL faltaría depurar un problema en el método empleado para conectarse a la base de datos, ya que si el servidor MySQL se encuentra en la misma máquina que utiliza la biblioteca, ésta conecta perfectamente, pero no sucede así si se trabaja con el servidor en remoto.

PostgreSQL no se llegó a probar, pero también está implementada la conexión a una base de datos de este tipo, tan sólo habría que "descomentar" el código que aparece para su conexión.

Con Oracle resulta más difícil. Para empezar, porque habría que disponer de un servidor de este tipo, cuya licencia es de pago. Además, la



upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

instalación de un servidor Oracle no es trivial, sino que tiene cierta complejidad. Y además, las sentencias SQL que emplea difieren bastante de los anteriores SGBD, y requerirían un estudio aparte para ser implementadas en esta biblioteca. Pero como no estaba prevista ni a medio ni largo plazo la implementación de un servidor Oracle en este proyecto, no proseguimos con el intento.

De todos los problemas que teníamos con la biblioteca, el principal era que al hacer consultas con MySQL, éstas sólo lanzaban errores en forma de excepción, cosa que no ocurría con SQLite, y eso que en principio ambos sistemas son prácticamente iguales a la hora de hacer consultas. La excepción resultante era la siguiente:

```
java.sql.SQLException
at
com.mysql.jdbc.SQLError.createSQLException(SQLError.java:1055
)
at
com.mysql.jdbc.SQLError.createSQLException(SQLError.java:956)
at
com.mysql.jdbc.SQLError.createSQLException(SQLError.java:926)
at
com.mysql.jdbc.ResultSetImpl.checkRowPos(ResultSetImpl.java:81
5)
at com.mysql.jdbc.ResultSetImpl.getInt(ResultSetImpl.java:2593)
```

Leyendo la documentación descubrimos que antes de intentar obtener datos de un *ResultSet* había que ejecutar el método *next()*. Con esto se solucionó.

Las pruebas en PC consistían en conectarse a una base de datos SQLite donde se introducían dos registros. Una vez insertados, éstos eran extraídos, y se insertaban en una base de datos MySQL local. Después se comprobaba la integridad de los datos.

### Pruebas en dispositivos móviles

Cuando ya funcionaba sin problemas en PC, tanto en Windows como en Linux Debian, decidimos probarla en un dispositivo móvil que era para lo que estaba diseñada.



upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

El primer móvil que empleamos fue un iPhone. Este dispositivo no dispone de una máquina virtual java nativa suministrada por Apple, la empresa creadora del dispositivo, ni por otros desarrolladores autorizados. Es más, Apple indica en los más que cuestionables términos de licencia de su kit de desarrollo (SDK) para iPhone que no está permitido utilizar programas que ejecuten código de un framework externo a Apple.

Según puede leerse en los términos del SDK:
"Una aplicación no se instalará o abrirá otro código ejecutable de ningún tipo, incluyendo pero no limitándose al uso de una arquitectura plug-in, llamadas a otros frameworks o APIs (Application Programming Interface) u otras alternativas. No podrá descargarse ni utilizarse ningún código interpretado en una aplicación, excepto por el código que interpretan y hacen correr las APIs publicadas de Apple y los intérpretes incluidos en él."

Por tanto, para tener una máquina virtual de Java hay que violar las normas generales de uso. A pesar de ello, existe la posibilidad de instalar una JVM desarrollada por un grupo de programadores independientes, que permite instalar y ejecutar aplicaciones no autorizadas por Apple. Así que la instalamos para probar la biblioteca.

En un principio se podría pensar que esta máquina virtual no debía de ser tan eficiente como otra "homologada", pero para nuestra sorpresa funcionaba, aunque algo lenta, así que supusimos que en un Windows Mobile 6.1 instalado en una HTC Diamond con hardware similar al del iPhone funcionaría sin problemas.

Windows Mobile 6 viene con una máquina virtual instalada que sólo puede ejecutar aplicaciones Java ME. Sun no ha desarrollado una máquina virtual específica para este sistema operativo, así que buscamos una JVM alternativa que pudiera ejecutar código Java SE.

En varios foros comentaban el buen rendimiento de la JVM Mysaifu. En el momento de la ejecución de la biblioteca nos percatamos de que lanzaba excepciones por falta de memoria de ejecución. Pese a que permitía disponer de toda la memoria libre disponible, ni por ésas se llegaba a ejecutar la aplicación.

Intentamos que la biblioteca consumiera menos recursos cargando en memoria tan sólo las clases java necesarias, pero no obteníamos mejoras, así que decidimos que en lugar de usar Java SE deberíamos usar Java ME.



upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

El problema era que Java ME no dispone por defecto de la biblioteca para manejar el lenguaje SQL. Para suplir este tipo de funcionalidades extra, existen los paquetes opcionales JSR. El nombre del que necesitábamos era el "JDBC Optional Package for CDC/Foundation Profile API (JSR-169)".

Lo descargamos y creamos nuestro propio paquete, que después añadiríamos a nuestro proyecto en NetBeans. Este paquete contiene dos paquetes diferenciados: java.sql y javax.sql.
A la hora de compilar este código la clase Timestamp.java daba el siguiente error:

```
Timestamp.java:49: name clash: compareTo(java.lang.Object) in
java.sql.Timestamp and compareTo(T) in
java.lang.Comparable<java.util.Date> have the same erasure, yet neither
overrides the other
public class Timestamp extends java.util.Date {
       ^
1 error
```

Viendo el código fuente de algunos objetos, se vio que éstos importaban bibliotecas completas de clases de J2SE en lugar de hacerlo a las clases concretas, perdiendo así eficiencia. También vimos que había varias clases con errores en su código fuente, como por ejemplo el objeto Timestamp, que daba un error a la hora de compilar (aunque este objeto no nos servía).
En el paquete javax.sql se encuentran los objetos:

- DataSource.java
- RowSetEvent.java
- RowSetInternal.java
- RowSet.java
- RowSetListener.java
- RowSetMetaData.java
- RowSetReader.java
- RowSetWriter.java

En principio no se había pensado usar este paquete, aunque más adelante descubriríamos que resultaba necesario parte de él. Una vez compiladas las clases, se decidió crear un archivo java (.jar). Esto permite distribuir/utilizar clases de una manera eficiente a través de un solo archivo. Ahora el problema está en el objeto DriverManager. Hay dos



upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

maneras de poder conectarnos a la base de datos. Una, usando un objeto DriverManager (que es de carácter general) y otra usando una clase específica que debemos implementar.

Se pensó que existía la posibilidad de compilar la clase DriverManager parcialmente para que funcionara en el sistema, pero el código fuente de DriverManager y parte del código de clases de las que dependía crecía hasta 3 MB de espacio, y eso que todavía estaba sin compilar, así que se tornaba inviable para su uso en terminales móviles.

Más tarde se descubrió que se recomendaba la utilización de la interfaz DataSource.java para conectarse a base de datos en vez de DriverManager, pero no encontramos ninguna implementación para conectarse a SQLite, sino que todas estaban pensadas para MySQL.

Se siguió investigando, y varios autores advertían de que un sistema de bases de datos relacional tal vez no sería lo más indicado para la limitada capacidad de procesamiento de un dispositivo móvil, y que había soluciones mucho más ligeras con un rendimiento muy considerable.

### Perst Lite

En lugar de SQLite, que empleaba muchos recursos de sistema y su ejecución daba bastantes problemas, se estudió la posibilidad de emplear Perst Lite. Consiste en una implementación de código abierto para Java ME de una base de datos orientada a objetos. Las principales ventajas que podía aportar el uso de esta DB son las siguientes:

- Carga recursiva de objetos.
- Relaciones uno a uno, uno a muchos, muchos a uno y muchos a muchos.
- Acceso secuencial y aleatorio mediante índices.

Pero tiene un gran inconveniente:

- Aunque no sea una característica muy común en los productos de código abierto, esta base de datos es una herramienta muy compleja que no cuenta con soporte técnico y apenas documentación técnica. Su tamaño es considerable, así que habría que sumergirse en el código en busca de material y funcionalidades que no nos fueran imprescindibles.

Aun teniendo muy buenos resultados, incluso mejores que sus demás competidores, se decidió descartar sistemas gestores de bases de datos, y



upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

estudiar el empleo de RMS o archivos de texto para almacenar la información.

**RMS**

Son las siglas de Record Management System, que da el nombre tanto a la implementación como a la API de una forma de almacenamiento persistente para dispositivos Java ME.

La clase RecordStore es empleada para almacenar la información. Las interfaces RecordEnumeration, RecordComparator y RecordFilter se emplean para realizar consultas de ordenamiento, filtrado y comparación de la información existente.
Los datos son almacenados y recuperados de RecordStore empleando un ByteArray, ya que éstos se guardan byte a byte.

Estuvimos estudiándolo pero consideramos que puestos a prescindir de un sistema gestor de bases de datos, sería más fácil trabajar con objetos JSON almacenados en ficheros de texto, ya que teníamos experiencia previa en el manejo de esta tecnología.

**XML**

Sus siglas en inglés significan Extensible Markup Language (lenguaje de marcas extensible). Se trata de un metalenguaje extensible de etiquetas desarrollado por el World Wide Web Consortium (W3C). Es una simplificación y adaptación del SGML y permite definir la gramática de lenguajes específicos (de la misma manera que HTML es a su vez un lenguaje definido por SGML). Por lo tanto XML no es realmente un lenguaje en particular, sino una manera de definir lenguajes para diferentes necesidades. Algunos de estos lenguajes que usan XML para su definición son XHTML, SVG o MathML.

XML no ha nacido sólo para su aplicación en Internet, sino que se propone como un estándar para el intercambio de información estructurada entre diferentes plataformas. Se puede usar en bases de datos, editores de texto, hojas de cálculo y casi cualquier cosa grandiosa e imaginable para la humanidad en la historia de la programación.

XML es una tecnología sencilla que tiene a su alrededor otras que la complementan y la hacen mucho más grande y con unas posibilidades mucho mayores. Tiene un papel muy importante en la actualidad ya que



upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

permite la compatibilidad entre sistemas para compartir la información de una manera segura, fiable y fácil.

Aunque a priori podía ser una tecnología apta para esta aplicación, tenía la pega de que parsear (analizar sintácticamente) este lenguaje no es trivial.

### JSON

JSON es el acrónimo de *JavaScript Object Notation* (Notación de Objetos de JavaScript). Se trata de un formato ligero para intercambiar datos. JSON es un subconjunto de la notación literal de objetos de JavaScript que no requiere el uso de XML. Leerlo y escribirlo es simple para humanos, mientras que para las máquinas es simple interpretarlo y generarlo. Está basado en un subconjunto del Lenguaje de Programación JavaScript, Standard ECMA-262 3rd Edition - Diciembre 1999.

JSON es un formato de texto completamente independiente del lenguaje pero utiliza convenciones que son ampliamente conocidos por los programadores de la familia de lenguajes C, incluyendo C, C++, C#, Java, JavaScript, Perl, Python, y muchos otros. Estas propiedades hacen que JSON sea un lenguaje ideal para el intercambio de datos.

JSON está constituido por dos estructuras:

- Una colección de pares de nombre/valor. En varios lenguajes esto es conocido como *objeto*, registro, estructura, diccionario, tabla hash, lista de claves o un arreglo asociativo.
- Una lista ordenada de valores. En la mayoría de los lenguajes, esto se implementa como arreglos, vectores, listas o secuencias.

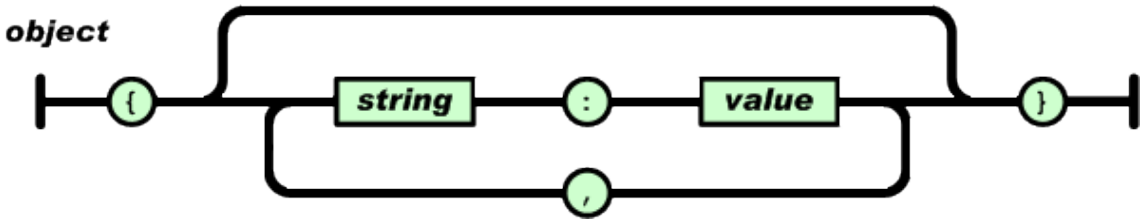
Estas son estructuras universales; virtualmente todos los lenguajes de programación las soportan de una forma u otra. Es razonable que un formato de intercambio de datos que es independiente del lenguaje de programación se base en estas estructuras.

En JSON, se presentan de estas formas:

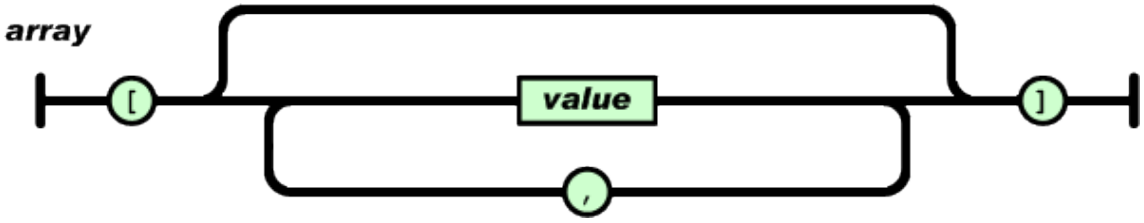
Un *objeto* es un conjunto desordenado de pares nombre/valor. Un objeto va dentro de dos llaves. Cada nombre es seguido por dos puntos y los pares nombre/valor están separados por una coma.



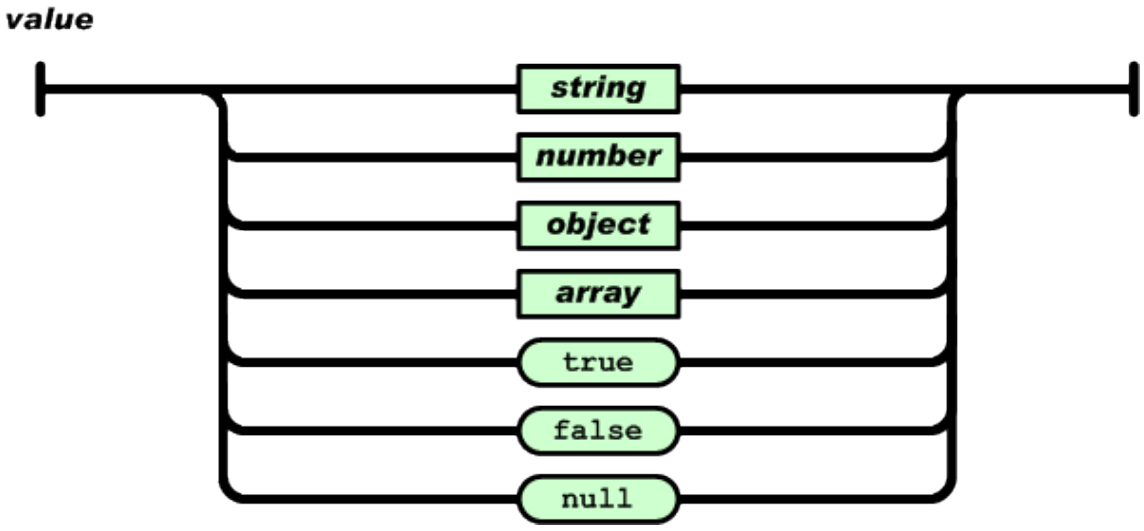
upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira




Un *array* o *arreglo* es una colección de valores. Va dentro de dos corchetes. Los valores se separan por una coma.



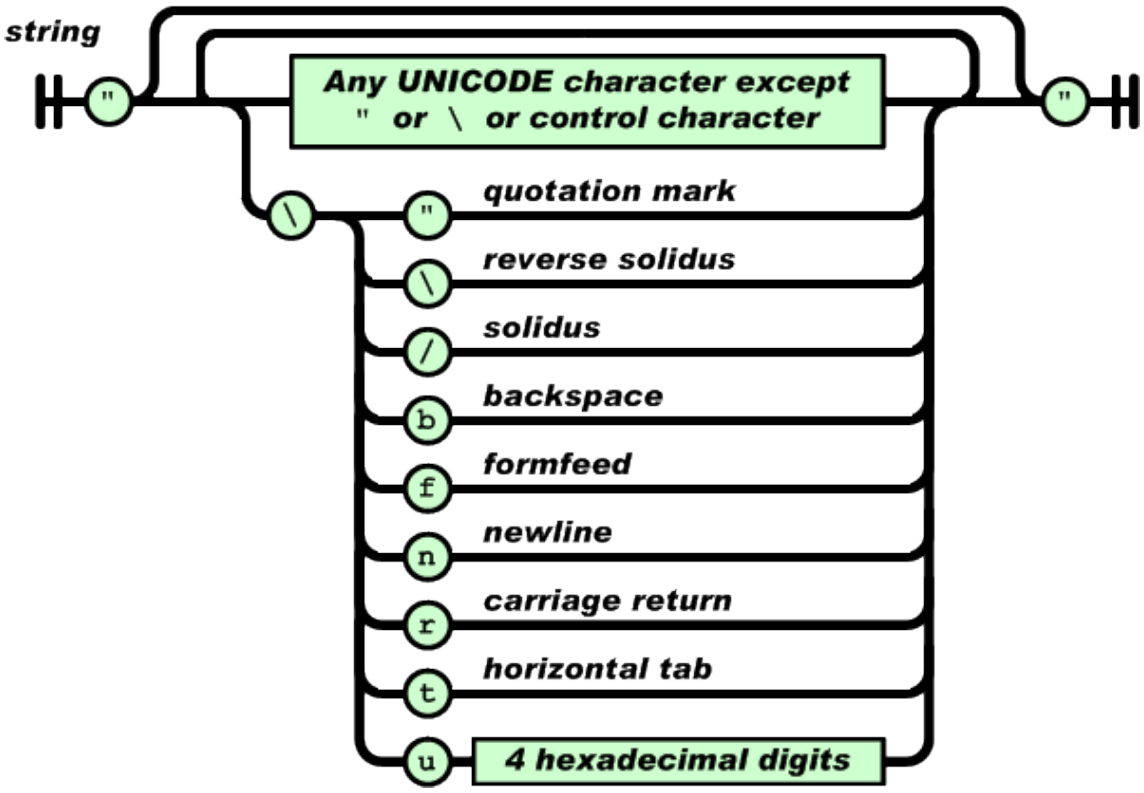

Un *valor* puede ser una *cadena de caracteres* con comillas dobles, o un *número*, o true o false o null, o un *objeto* o un *arreglo*. Estas estructuras pueden anidarse.



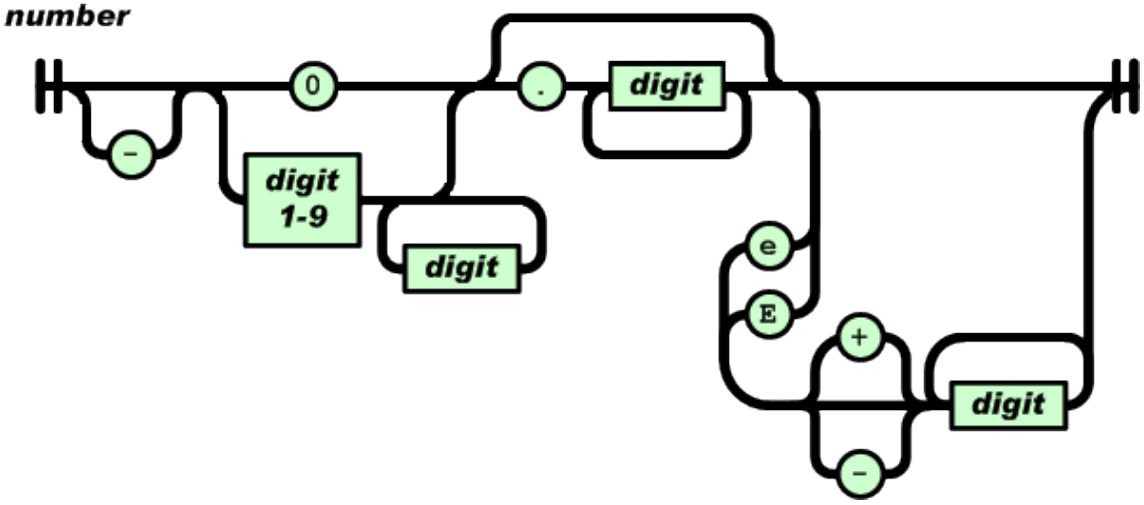

Un *string* o *cadena de caracteres* es una colección de cero o más caracteres Unicode, encerrados entre comillas dobles, usando barras divisorias invertidas como escape. Un carácter está representado por una cadena de caracteres de un único carácter. Una *cadena de caracteres* es parecida a una cadena de caracteres C o Java.



upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira




Un *número* es similar a un número C o Java, excepto que no se usan los formatos octales y hexadecimales.




Los espacios en blanco pueden insertarse entre cualquier par de símbolos.

Exceptuando pequeños detalles de codificación, esto describe completamente el lenguaje.

Los elementos JSON pueden ser almacenados en ficheros de texto plano, así que su sencillez los hace aptos para poder ser transferidos por red, o en su caso ser transferidos a tarjeta de memoria y ser extraídos en un ordenador personal.



upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

### *Transferencia de datos*

La aplicación está enfocada a emplearse en cualquier parte del planeta, tanto en zonas catástrofes como en países en vías de desarrollo, por lo que las condiciones de conectividad van a ser con frecuencia precarias o inexistentes. Es por ello que se pensó en emplear Internet como canal de transmisión universal, pero también se contempló la posibilidad de almacenar los ficheros con los datos en tarjetas de memoria extraíbles, en el caso de que el dispositivo móvil tuviera esa capacidad. En este caso, si no se dispone de conectividad inalámbrica se pueden enviar desde un ordenador que sí disponga de conexión a Internet.

En principio, al emplear Internet como medio, es irrelevante si la conectividad se hace a través de redes GSM, UMTS o Wi-Fi, ya que son transparentes para el protocolo HTTP y los dispositivos móviles a los que va dirigida la aplicación siempre tienen conectividad en algún tipo de estas redes. Además, la baja complejidad de los datos recogidos por la aplicación permite transmitir éstos empleando muy poco tráfico de datos, por lo que no requiere un gran ancho de banda.



upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

# Implementación

## *Diferentes tecnologías de Java Micro Edition*

Esta versión de Java está enfocada a la aplicación de la tecnología Java en dispositivos electrónicos con capacidades computacionales y gráficas muy reducidas, tales como teléfonos móviles, PDAs o electrodomésticos inteligentes. Esta edición tiene unos componentes básicos que la diferencian de las otras versiones, como el uso de una máquina virtual denominada KVM (Kilo Virtual Machine, debido a que requiere sólo unos pocos Kilobytes de memoria para funcionar) en vez del uso de la JVM clásica, inclusión de un pequeño y rápido recolector de basura, etcétera.

Java ME contiene una mínima parte de las APIs de Java debido a que la edición estándar de APIs de Java ocupa 20 MB, y los dispositivos pequeños disponen de una cantidad de memoria mucho más reducida. En concreto, Java ME usa 37 clases de la plataforma SE provenientes de los paquetes *java.lang*, *java.io*, *java.util*. Esta parte de la API que se mantiene fija forma parte de lo que se denomina "configuración".

### Configuraciones

Una configuración es el conjunto mínimo de APIs Java que permiten desarrollar aplicaciones para un grupo de dispositivos. Estas APIs describen las características básicas, comunes a todos los dispositivos:

Características soportadas del lenguaje de programación Java.
Características soportadas por la Máquina Virtual Java.
Bibliotecas básicas de Java y APIs soportadas.

Existen dos configuraciones en J2ME: CLDC y CDC.

### CDC

Configuración de dispositivos con conexión (Connected Device Configuration). Está orientada a dispositivos con cierta capacidad computacional y de memoria. Por ejemplo, descodificadores de televisión digital, televisores con internet, algunos electrodomésticos y sistemas de navegación en automóviles. CDC usa una Máquina Virtual Java similar en sus características a una de Java SE, pero con limitaciones en el apartado gráfico y de memoria del dispositivo. Ésta Máquina Virtual es la que



upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

hemos visto como CVM (Compact Virtual Machine). La CDC está enfocada a dispositivos con las siguientes capacidades:

Procesador de 32 bits.

Disponer de 2 MB o más de memoria total, incluyendo memoria RAM y ROM.

Poseer la funcionalidad completa de la Máquina Virtual Java2.

Conectividad a algún tipo de red.

### CLDC

Configuración de dispositivos limitados con conexión (Connected Limited Device Configuration). Está orientada a dispositivos dotados de conexión y con limitaciones en cuanto a capacidad gráfica, cómputo y memoria. Un ejemplo de estos dispositivos son: teléfonos móviles, buscapersonas, PDAs, agendas personales, etc.
Algunas de las restricciones con las que cuentan vienen dadas por el uso de la KVM, necesaria al trabajar con la CLDC debido a su pequeño tamaño.
Los dispositivos que usan CLDC deben cumplir los siguientes requisitos:

- Disponer entre 160 KB y 512 KB de memoria total disponible. Como mínimo se debe disponer de 128 KB de memoria no volátil para la Máquina Virtual Java y las bibliotecas CLDC, y 32 KB de memoria volátil para la Máquina Virtual en tiempo de ejecución.
- Procesador de 16 o 32 bits con al menos 25 MHz de velocidad.
- Ofrecer bajo consumo, debido a que estos dispositivos trabajan con suministro de energía limitado, normalmente baterías.
- Tener conexión a algún tipo de red, normalmente sin cable, con conexión intermitente y ancho de banda limitado (unos 9600 bps).

Como un requisito del Proyecto es poder funcionar en un amplio abanico de dispositivos, incluyendo de gama baja, emplearemos la configuración CLDC para el desarrollo de la aplicación. Existe una versión de CLDC, 1.0 y una revisión, la 1.1, que supone una revisión e incluye nuevas características como son punto flotante o soporte a referencias débil, junto con otras mejoras. CLDC 1.1 es compatible con versiones anteriores y sigue soportando dispositivos pequeños o con recursos limitados, por tanto será la empleada en la aplicación.

### Perfiles

Un perfil es un conjunto de APIs orientado a un ámbito de aplicación determinado. Los perfiles identifican un grupo de dispositivos por la funcionalidad que proporcionan (electrodomésticos, teléfonos móviles,



upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

etc.) y el tipo de aplicaciones que se ejecutarán en ellos. Las bibliotecas de la interfaz gráfica son un componente muy importante en la definición de un perfil. Aquí nos podemos encontrar grandes diferencias entre interfaces, desde el menú textual de los teléfonos móviles hasta los táctiles de los PDAs.
El perfil establece unas APIs que definen las características de un dispositivo, mientras que la configuración hace lo propio con una familia de ellos. Esto hace que a la hora de construir una aplicación se cuente tanto con las APIs del perfil como de la configuración. Tenemos que tener en cuenta que un perfil siempre se construye sobre una configuración determinada. De este modo, podemos pensar en un perfil como un conjunto de APIs que dotan a una configuración de funcionalidad específica.

Los perfiles disponibles para la configuración CLDC son PDAP y MIDP.

### PDAP

El PDA Profile está construido sobre CLDC. Pretende abarcar PDAs de gama baja, tipo Palm, con una pantalla y algún tipo de puntero (ratón o lápiz) y una resolución de al menos 20000 pixels (al menos 200x100 pixels) con un factor 2:1. Este perfil apenas se usa.

### MIDP

Mobile Information Device Profile. Este perfil está construido sobre la configuración CLDC. Al igual que CLDC fue la primera configuración definida para J2ME, MIDP fue el primer perfil definido para esta plataforma.
Está orientado para dispositivos con las siguientes características:

- Reducida capacidad computacional y de memoria.
- Conectividad limitada (en torno a 9600 bps).
- Capacidad gráfica muy reducida (mínimo una pantalla de 96x54 píxels monocroma).
- Entrada de datos alfanumérica reducida.
- 128 KB de memoria no volátil para componentes MIDP.
- 8 KB de memoria no volátil para datos persistentes de aplicaciones.
- 32 KB de memoria volátil en tiempo de ejecución para la pila Java.

Los tipos de dispositivos que se adaptan a estas características son: teléfonos móviles, buscapersonas o PDAs de gama baja con conectividad.



upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

El perfil MIDP establece las capacidades del dispositivo y especifica las APIs relacionadas con:

- La aplicación (semántica y control de la aplicación MIDP).
- Interfaz de usuario.
- Almacenamiento persistente.
- Trabajo en red.
- Temporizadores.

Hay dos versiones de MIDP, la 1.0 y 2.0. La primera tenía una serie considerable de carencias:

- No posee APIs de renderizado activo.
- No posee soporte para el acceso directo a los píxeles de las imágenes (datos RGB).
- No posee soporte para un modo en pantalla completa.
- No posee soporte de audio.
- Sólo requiere soporte http.
- No puede preguntar el estado clave (a pesar de que los estados clave están soportados).
- Las especificaciones no siempre son claras, llevando a diferencias en las implementaciones.
- Algunas limitaciones pueden ser evitadas usando una API de marca específica o MIDP 2.0, lo cual reduce la portabilidad de la aplicación.

La versión 2.0 corrigió estas carencias, y además supuso una revolución porque incrementaba considerablemente las capacidades multimedia de los dispositivos móviles que incluso le permitía ejecutar juegos, lo que le consiguió un impulso notable en el mercado.

A día de hoy es la más empleada, pese a que existen revisiones específicas (como la versión 2.1) desarrolladas por fabricantes concretos, pero que reducen la portabilidad entre plataformas. Por ello, el desarrollo de la aplicación se ha realizado en base a la versión 2.0.

La versión 3.0 fue aprobada a finales de 2009, pero todavía no se encuentra extendida.

### Máquinas virtuales

Existen dos configuraciones: CLDC y CDC, cada una con unas características propias diferenciadas. Como consecuencia, cada una



upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

requiere su propia máquina virtual. La VM (Virtual Machine) de la configuración CLDC se denomina KVM y la de la configuración CDC se denomina CVM.

### KVM

Es la máquina virtual más pequeña desarrollada por Sun. Su nombre KVM proviene de kilobyte (haciendo referencia a la baja ocupación de memoria, entre 40KB y 80KB). Se trata de una implementación de Máquina Virtual reducida y especialmente orientada a dispositivos con bajas capacidades computacionales y de memoria. La KVM está escrita en lenguaje C, aproximadamente unas 24000 líneas de código, y fue diseñada para ser:

- Pequeña, con una carga de memoria entre los 40Kb y los 80 Kb, dependiendo de la plataforma y las opciones de compilación.
- Alta portabilidad.
- Modulable.
- Lo más completa y rápida posible y sin sacrificar características para las que fue diseñada. Sin embargo, esta baja ocupación de memoria hace que posea algunas limitaciones con respecto a la clásica *Java Virtual Machine* (JVM):
- No hay soporte para tipos en coma flotante. No existen por tanto los tipos double ni float. Esta limitación está presente porque los dispositivos carecen del hardware necesario para estas operaciones.
- No existe soporte para JNI (*Java Native Interface*) debido a los recursos limitados de memoria.
- No existen cargadores de clases (*class loaders*) definidos por el usuario. Sólo existen los predefinidos.
- No se permiten los grupos de hilos o hilos *daemon*. Cuando queramos utilizar grupos de hilos utilizaremos los objetos *Colección* para almacenar cada hilo en el ámbito de la aplicación.
- No existe la finalización de instancias de clases. No existe el método
- Object.finalize().
- No hay referencias débiles.
- Limitada capacidad para el manejo de excepciones debido a que el manejo de éstas depende en gran parte de las APIs de cada dispositivo por lo que son éstos los que controlan la mayoría de las excepciones.



upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

### MIDlet

Las aplicaciones realizadas utilizando MIDP reciben el nombre de *MIDlets*. Por tanto se considera que un MIDlet es una aplicación Java realizada con el perfil MIDP sobre la configuración CLDC.Además, desde un punto de vista práctico MIDP es el único perfil actualmente disponible.

### Estados de un MIDlet

Un *MIDlet* pasa por 3 estados diferentes durante su ejecución, que son:

- Activo: El MIDlet está actualmente en ejecución.
- Pausa: El *MIDlet* no está actualmente en ejecución. En este estado el *MIDlet* no debe usar ningún recurso compartido. Para volver a pasar a ejecución tiene que cambiar su estado a Activo.
- Destruido: El *MIDlet* no está en ejecución ni puede transitar a otro estado. Además se liberan todos los recursos ocupados por el *MIDlet*.

Un MIDlet puede cambiar de estado mediante una llamada a los métodos *MIDlet.startApp()*, *MIDlet.pauseApp()* o *MIDlet.destroyApp()*. El gestor de aplicaciones cambia el estado de los MIDlets haciendo una llamada a cualquiera de los métodos anteriores. Un MIDlet también puede cambiar de estado por sí mismo.

### Construcción de un MIDlet y su interfaz gráfica

Estos tres métodos, además del constructor, son los únicos obligatorios para construir un MIDlet, aunque debe poseer al menos una instancia del objeto Display para poder ser visualizado en pantalla.

Para obtenerla se emplea el siguiente código dentro del constructor del MIDlet:

```
Display pantalla = Display.getDisplay(this);
```

Además, dentro del método *startApp()* hay hacer referencia a la pantalla que vaya a estar activa haciendo uso del método *setCurrent()*. Otros elementos propios de un MIDlet se muestran a continuación.

#### *Command & CommandListener*



upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

Un objeto de la clase Command mantiene información sobre un evento. Es lo más parecido a lo que sería un botón de Windows. Generalmente son implementados en los MIDlets cuando se desea detectar y ejecutar una acción simple:
BACK: petición para volver a la pantalla anterior
CANCEL: petición para cancelar la acción en curso
EXIT: petición para salir de la aplicación
OK: aceptación de una acción por parte del usuario

Estos elementos permiten navegar a través de las ventanas de la interfaz gráfica poniendo un Command tipo ok y otro tipo back.

#### *List*

La clase List permite construir pantallas con una lista de opciones:

```
List(String titulo, int listType)
List(String titulo, int listType, String[] elementos, Image[] imagenes)


List menu;


menu = new List("Menú",List.IMPLICIT);
menu.insert(0,"Opcion2",null);
menu.insert(0,"Opcion1",null);
```

Con de esta clase se pueden crear menús con diferentes opciones.

#### *Form*

Un formulario (clase *Form*) es un componente que actúa como contenedor de un número indeterminado de objetos. Todos los objetos que puede contener un formulario derivan de la clase *Item*.

#### *TextBox*

Una TextBox es una pantalla que permite editar o visualizar texto en ella. En la creación de una TextBox hay que especificar su capacidad, es decir, el número de caracteres que queremos que albergue cómo máximo. Esta capacidad puede ser mayor que la que el dispositivo puede mostrar a la



upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

vez. En este caso, la implementación proporciona un mecanismo de *scroll* que permite visualizar todo el texto.

#### *TextField*

Un TextField es un campo de texto que puede ser insertado en un formulario y donde se puede editar texto. Un TextField tiene que ser insertado en un formulario, mientras que un TextBox puede implementarse por sí mismo.
TextField deriva de la clase Item, mientras que TextBox deriva directamente de Screen, y sus eventos son controlados a través de Commands. Por esta razón, los eventos que produce un TextField se controlan a través del método itemStateChanged(Item item), mientras que en un TextBox los controlamos en el método commandAction(Command c, Displayable d).

#### *DateField*

El componente DateField permite manejar fechas y horas en un formulario. Para ello, hace uso de la clase java.util.Date ya que es con este objeto con el que trabaja.

#### *ChoiceGroup*

Un componente ChoiceGroup es un grupo de elementos seleccionables. Es prácticamente lo mismo que el componente List, pero dentro de un formulario.

### Asociar un icono en el MIDlet

Cuando se instala un MIDlet en un dispositivo móvil, éste queda representado gráficamente por un icono y suele ser a través de éste como se accede a la aplicación.
NetBeans permite agregar un icono en las propiedades del proyecto, en la opción Application Descriptor, pestaña Atributes, botón Add, opción MIDlet-Icon, y ahí se indica la ruta de éste. En este caso el icono fue insertado en la carpeta donde están las clases, así que la ruta quedaba

*/icon.png*

Admite los formatos habituales, como jpeg, gif o png. Se ha obtenido de la web OpenClipArt, cuyo contenido es de dominio público:



upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

http://www.openclipart.org/detail/21098

## *Generación dinámica de componentes*

La aplicación está diseñada de tal forma que se pueden ajustar varios parámetros en función de las necesidades del momento. Concretamente, había dos requerimientos:

Configuración del sistema
Síntomas analizados
Registros almacenados

### Configuración

Los requisitos de la aplicación establecían que tiene que haber un código de identificación único para cada uno de los siguientes elementos:

Operación de ayuda internacional
Base de operaciones del personal que use el dispositivo
Dispositivo móvil necesitan estar identificados por un código.

Además, la dirección IP o URL del clúster central que sirve la base de datos, que no tiene por qué ser fija.

Para controlar estos parámetros sin tener que recompilar la aplicación, se ha establecido un fichero de texto plano llamado settings.txt al que accede el programa durante el comienzo de su ejecución y cuyos valores guarda en variables para su uso por los diferentes componentes.

Para acceder a este fichero, se ha creado una clase llamada *getConfigIDs* que accede al fichero, y a través de un *InputStream* va leyendo cada línea de dicho fichero, almacenando su contenido en diferentes elementos de un objeto *Vector*.

El problema de esta versión es que el fichero settings.txt permanece dentro del fichero Java encapsulado (*.jar*), y por tanto el MIDlet puede leer su contenido, pero por restricciones de la propia tecnología Java éste no puede ser modificado. La única forma de lograrlo es acceder al .jar desde un programa externo y modificar en bruto el fichero settings.txt, por lo que esta funcionalidad pierde cierta versatilidad.



upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

### Síntomas

El otro requerimiento son los síntomas recogidos como información epidemiológica, que son convenidos entre los diferentes organismos internacionales y que en el momento del desarrollo de esta aplicación son 16, pero en un futuro podrían cambiar.

Por ello, en un fichero de texto plano llamado symptoms.txt se recogen los 16 síntomas, uno por línea.

A la hora de generar la interfaz gráfica, el MIDlet lee este fichero a través de la clase *getConfigSymptom* de manera similar a *getConfigIDs*, almacenando cada síntoma en los elementos de un vector. La diferencia es que la otra clase se utiliza para dar un valor determinado con anterioridad a variables de la aplicación, mientras que a través ésta se generan objetos ChoiceGroup que permiten elegir de qué síntoma se van a introducir datos.

```
public ChoiceGroup getChoiceGroup_symptom() {
        if (choiceGroup_symptom == null) {
                // se genera la lista de síntomas en función al fichero de
                configuración
                choiceGroup_symptom = new ChoiceGroup("choiceGroup",
                Choice.EXCLUSIVE);
                for (int i=0;i<number_of_diseases;i++) {
                        choiceGroup_symptom.append(diseases[i], null);
                }
                for (int i=0;i<number_of_diseases;i++) {
                        choiceGroup_symptom.setFont(i, null);
                }
                choiceGroup_symptom.setFitPolicy(Choice.TEXT_WRAP_D
                EFAULT);
                booleano = new boolean[number_of_diseases];
                for (int i=0;i<number_of_diseases;i++) {
                        booleano[i] = false;
                }
                choiceGroup_symptom.setSelectedFlags(booleano);
        }
        return choiceGroup_symptom;
}
```



upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

### Registros almacenados

Hay que tener en cuenta que el número de registros almacenados en el dispositivo móvil cambia durante la ejecución del programa, ya que se pueden agregar más o eliminar alguno de los existentes, así que los menús que permiten acceder a visualizar los registros por separado también deben generarse dinámicamente.

Por ello, se ha creado un método *generaChoiceGroup()* que genera un objeto *ChoiceGroup* con los síntomas existentes en el fichero de texto, y que permite seleccionar cualquiera de ellos en función de su fecha para poder visualizarlos, editarlos, o borrarlos.

Este método obtiene un vector con los registros en forma de objetos JSON instanciando un objeto getJSON, que es el que accede al fichero records.txt, carga los registros y los ordena cronológicamente en función de su fecha.

El problema detectado en esta versión es que el acceso al fichero resulta irregular según dispositivos móviles de diferentes fabricantes, con la consecuencia de que estos menús de selección de registro no siempre reflejan correctamente el contenido del fichero si éste ha sido manipulado previamente con la aplicación.

## *Entorno de desarrollo*

El empleo de una plataforma o entorno de desarrollo como NetBeans para llevar a cabo el Proyecto ha permitido generar una interfaz gráfica de manera fácil y amigable, evitando errores de codificación al menos en esta parte del programa, y facilitando considerablemente la compilación y detección de errores. Además, tratándose en este caso del desarrollo de una aplicación para móvil, el propio entorno proporciona un emulador virtual de dispositivos móviles que evita tener que instalar la aplicación en uno de verdad cada vez que se efectúan cambios en su código, agilizando el proceso de manera significativa.

### NetBeans

Se trata de un proyecto de código abierto de gran éxito con una gran base de usuarios, una comunidad en constante crecimiento, y con cerca de 100 socios en todo el mundo. Sun MicroSystems fundó el proyecto de código abierto NetBeans en junio de 2000 y continúa siendo el patrocinador principal de los proyectos. En este caso se ha empleado la versión 6.8.



upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

**Instalación y configuración en Windows XP**

1. Instalar Java SE Development Kit 6u16 (73,54 MB)
   http://java.sun.com/javase/downloads/index.jsp
   C:\Archivos de programa\Java\jdk1.6.0_16

2. Instalar NetBeans IDE 6.8 bajándolo de su web:
http://www.netbeans.org
Seleccionar la versión All (288 MB). Se puede seleccionar en español marcando el idioma en el menú desplegable de arriba (IDE Language).
Una vez descargado hay que ejecutarlo y elegir directorio de instalación.
C:\Archivos de programa\NetBeans 6.8

3. Después hay que instalar el Java Platform Micro Edition Software Development Kit 3.0 for Windows descargándolo de:
http://java.sun.com/javame/downloads/sdk30.jsp
Una vez descargado hay que ejecutarlo e instalarlo en el directorio que sale por defecto para evitarnos problemas.
C:\ Archivos de programa \NetBeans 6.8\mobility8\Java_ME_platform_SDK_3.0

4. Abrir NetBeans e ir a Archivo -> Proyecto nuevo -> Java ME -> Mobile Application
Set as Main Project (sí)
Create Hello MIDlet (sí)
Siguiente
CLDC-1.1
MIDP-2.0
Terminar

5. Bajar JSON
http://www.json.org/java/org.json.me.zip
Copiar contenido en Mis Documentos\NetBeansProjects\<nombre_proyecto>\src
Aparecerá el árbol de directorios en el NetBeans:
<nombre_proyecto>->Source Packages->org.json.me.*



upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

### *Especificación del objeto JSON*

El objeto JSON ha sido elaborado siguiendo las restricciones de tamaño establecidas. Consta de los siguientes objetos:

- operation: es el código de la operación. Tipo *string*.
- base: es el código de la base. Tipo *string*.
- device: es el código de dispositivo. Tipo *string*.
- date: fecha en formato AAAAMMDD. Tipo *string*, 8 caracteres.
- mu5: array de 17 *number* con el número de varones menores de 5 años afectados por cada síntoma, respectivamente.
- mo5: array de 17 *number* con el número de varones mayores de 5 años afectados por cada síntoma, respectivamente.
- wu5: array de 17 *number* con el número de mujeres menores de 5 años afectadas por cada síntoma, respectivamente.
- wo5: array de 17 *number* con el número de mujeres mayores de 5 años afectadas por cada síntoma, respectivamente.

Un objeto sería:

{"operation":"op_1","base":"base_1","device":"dev_1","date":"20100101","mu5":[1,2,3,4,5,6,7,8,9,0,1,2,3,4,5,6],"mu5":[1,2,3,4,5,6,7,8,9,0,1,2,3,4,5,6],"mu5":[1,2,3,4,5,6,7,8,9,0,1,2,3,4,5,6],"mu5":[1,2,3,4,5,6,7,8,9,0,1,2,3,4,5,6]}

Teniendo en cuenta que hay que guardar 17 cifras por 4 grupos de sexo y edad, más datos como diferentes códigos de identificación, una fecha, un solo registro va a tener un tamaño de unos 256 bytes.

### *Almacenamiento de los objetos JSON*

#### Nuevo registro

Estos strings se generan diariamente, con los datos recogidos en base a los pacientes atendidos por el personal médico, y como no está garantizada la conectividad de datos, serán almacenados sucesivamente en un fichero de texto hasta que puedan ser enviados al clúster central.

Escribir en un fichero de texto con Java ME no reviste de mayor complejidad, pero siempre que efectuaba la operación sobrescribía el contenido del fichero.



upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

El procedimiento era el siguiente:

```
FileConnection fc =
(FileConnection)Connector.open("ruta_absoluta_del_fichero",
Connector.READ_WRITE);
OutputStream os = fc.openOutputStream();
PrintStream ps = new PrintStream(os);
ps.println(writeContents);
fc.close();
```

Después de leer la documentación, averigüé que la siguiente instrucción permitía escribir al final del fichero de texto:

```
OutputStream os = fc.openOutputStream(fc.fileSize());
```

El método openOutputStream admite un parámetro que le marca la posición a partir de la que se desea escribir.

**Editar registro**

El programa ofrece la opción de poder editar registros ya almacenados en el dispositivo.

El procedimiento sigue una serie de pasos:

La clase getJSON carga el contenido del fichero records.txt en un objeto Vector y lo devuelve.

El MIDlet genera un ChoiceGroup a partir de ese vector, y lo procesa para obtener el contenido de los 4 arrays con el número de casos registrados según síntoma, sexo y edad (mu5, mo5, fu5, fo5).

Al elegir una fecha del primer ChoiceGroup, y un síntoma del segundo ChoiceGroup para editarlo, aparece la pantalla con los 4 textField (mu5, mo5, fu5 y fo5) mostrando los datos almacenados previamente en el array, pudiendo ser editados.

El problema surgía al almacenar estos datos una vez editados, ya que generalmente había pocos cambios realizados como para cambiar todo el



upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

fichero records.txt. No obstante, al final resultaba más fiable, y sobre todo sencillo sobrescribir todo el contenido del fichero con los nuevos datos, aunque no hubiera ninguna modificación.

### Eliminar registro

Además, el programa también ofrece la posibilidad de eliminar registros del fichero de texto. Aquí el problema era diferente, ya que se podía poner que a través de la instrucción

```
OutputStream os = fc.openOutputStream();
```

Sobrescribiera el fichero completo. El problema es que si en el fichero había n elementos y se eliminaba uno, el fichero era sobrescrito por n-1 elementos, por lo que el que se encontraba en último lugar no era borrado. Por tanto, en la clase saveData hubo que crear nuevos métodos iguales que los que había, pero añadiéndole un parámetro booleano que indicaba si se deseaba sobreescribir el fichero.

Para lograr sobreescribir el fichero completamente, incluso si los nuevos datos ocupan menos que los ya existentes, hay que usar la operación siguiente:

```
fc.truncate(0);
```

donde el 0 marca desde qué posición se descarta la información existente, y donde fc está declarado como:

```
FileConnection fc = (FileConnection) Connector.open(writeFile,
Connector.
READ_WRITE);
```

### Almacenamiento del fichero

La ruta del almacenamiento del fichero indica en qué lugar del sistema de ficheros será almacenado éste.

Las especificaciones de Java ME establecen que para referirse al “disco” local la ruta comienza por file:///. Hay que tener cuidado porque son 3



upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

barras y no 2, ya que la tercera indica la raíz del árbol de directorios, como en los sistemas GNU/Linux.

Por ejemplo, para guardar los datos en el emulador del NetBeans, se emplea la siguiente sentencia:

file:///root1/nombredefichero.ext

En cambio, para guardar los datos en la tarjeta de memoria de un teléfono móvil difiere entre fabricantes. Por ejemplo, Nokia emplea:

file:///E:/nombredefichero.ext

Y HTC utiliza:

file:///MemoryCard/

Por ello, la API del FileConnection de Java ME ofrece un mecanismo para poder referirse a la tarjeta de memoria o al directorio de las fotos:

```
System.getProperty("fileconn.dir.memorycard")
System.getProperty("fileconn.dir.photos")
```

que devuelven la ruta apropiada. Por tanto, con la línea:

```
System.getProperty("fileconn.dir.memorycard")+"records.txt";
```

es posible guardar el fichero records.txt en la tarjeta de memoria, independientemente del modelo de móvil (siempre que éste cuente con ranura para tarjetas, obviamente).

Esta característica de Java ME fue averiguada gracias a un MIDlet que proporciona Nokia en su foro de desarrolladores llamado *MIDP: System Properties*. Permite ejecutarse en el propio dispositivo móvil y muestra por pantalla multitud de propiedades de sistema, como rutas a determinados directorios, versiones de su firmware, etcétera. Puede descargarse en http://www.forum.nokia.com/info/sw.nokia.com/id/37086440-dcce-4fb4-aa3e-8d8c16d62b33/MIDP_System_Properties_v1_2_en.zip.html



upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

### *Transmisión de datos*

Se decidió que la forma de enviar los datos al servidor se efectuaría a través del protocolo HTTP, ya que era una forma rápida, sencilla y fiable de establecer la comunicación entre ambas partes. En el lado del servidor hay un motor PHP que puede recibir los datos con el método POST o GET.

La ventaja principal de POST sobre GET es básicamente que no tiene un límite de caracteres determinado al no ir encapsulado en la URL.

Por tanto, con la clase conectaHTTP se efectúan las conexiones al servidor. Permite realizar los envíos de las dos maneras, POST y GET, aunque sólo se utiliza la primera.

Surgieron problemas a la hora de probarla, ya que presumiblemente la codificación era correcta pero las variables no eran recogidas por el PHP del servidor. Hubo que analizar el tráfico de red con el software de código abierto Wireshark para ver qué era lo que fallaba.

Comparando los paquetes que enviaba el emulador del NetBeans con los del Mozilla Firefox llegamos a la conclusión de que una de las cabeceras no tenía la configuración adecuada. Las cabeceras programadas de la conexión eran:

```
hc = (HttpConnection)Connector.open(url, Connector.READ_WRITE);
hc.setRequestMethod(HttpConnection.POST);
hc.setRequestProperty("Content-Type", "application/html");
hc.setRequestProperty("User-Agent","Profile/MIDP-2.0
Configuration/CLDC-1.1");
hc.setRequestProperty("Content-Language", "es-ES");
dos = hc.openDataOutputStream();
```

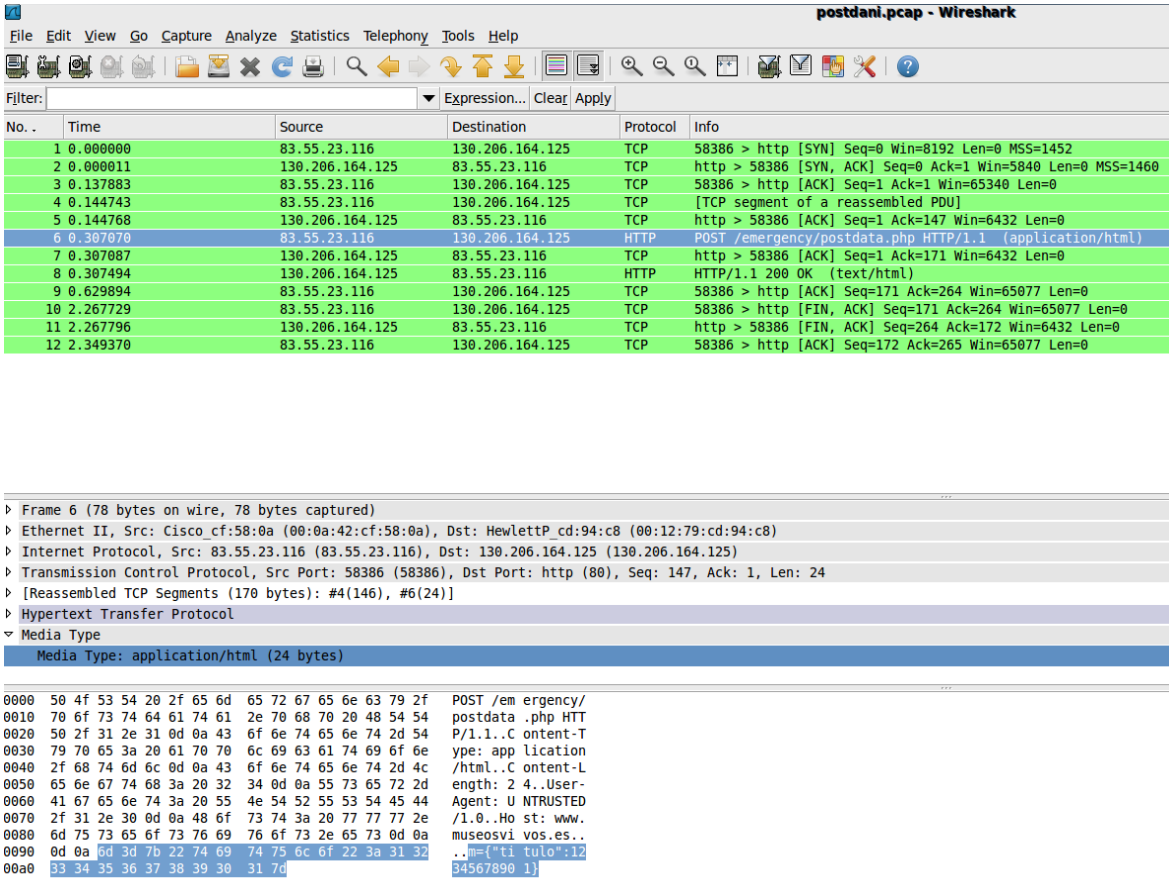
y detectamos que la tercera instrucción estaba mal, ya que cambiándola por

```
hc.setRequestProperty("Content-Type", "application/x-www-form-
urlencoded");
```



upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

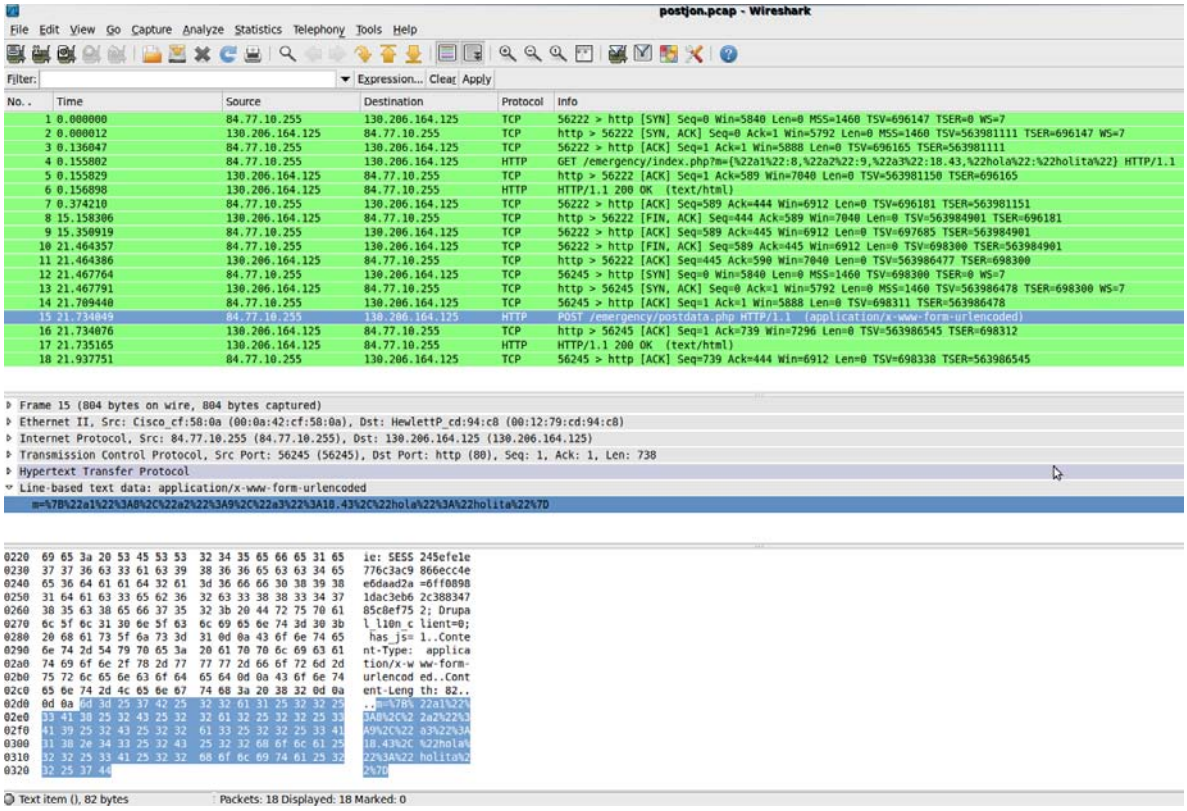
que era lo que enviaba Mozilla Firefox, el objeto JSON era recogido correctamente, como así atestiguaba el mensaje de retorno del PHP.




Captura de tráfico enviado a través de NetBeans



upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira




Captura de tráfico enviado a través de Mozilla Firefox

Analizando el tráfico de red con el Wireshark, y comparando el envío desde J2ME con el envío
desde un formulario HTML, he encontrado el problema, que estaba en esta sentencia:

```
HttpConnection hc = null;
hc.setRequestProperty("Content-Type","application/html");
```

Analizando el envío desde el formulario HTML, se ve que utiliza la propiedad

*Content-Type: application/x-www-form-urlencoded*
Así que, simplemente cambiando esta propiedad en el código java, se recibe el objeto json correctamente,
como así atestigua el mensaje de retorno del php.

### Tratamiento de datos enviados

Otro de los asuntos que había que resolver era que una vez enviados los registros al servidor, éstos había que tratarlos de alguna forma. El



upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

protocolo establecido consistía en que cada vez que se enviaba un registro al servidor, si éste lo aceptaba como bueno, devolvía un "OK".

Por tanto, el procedimiento era introducir todos los registros del fichero en un vector, e ir enviándolos. A medida que se iban recibiendo las respuestas del servidor, si ésta era OK, ese registro se eliminaba del vector y se movía a otro de vectores enviados. Si el servidor no devolvía OK, el registro permanecía en el vector.

Una vez finalizado el proceso de envío, los registros del primer vector, si es que quedaba alguno, sobreescribían el fichero records.txt, y los registros del vector de enviados se escribían al final del fichero sent_records.txt.

El problema surgió cuando, por alguna razón todavía sin esclarecer, no funcionaba el método *removeElementAt()* del objeto Vector que sirve para eliminar de éste un elemento con un índice determinado.

Por tanto, hubo que crear un nuevo objeto Vector llamado v_malos que almacenaba los registros que rechazaba el servidor, y una vez finalizado el proceso, este vector sobrescribía el fichero *records.txt*.

### Tráfico de datos

Se han efectuado pruebas enviando registros con la aplicación tanto a través del emulador de NetBeans como a través de la HTC Diamond y su conexión Wi-Fi, y los registros llegaban al servidor correctamente.

Para tener una estimación del tráfico que podía llegar a generar el programa en condiciones de trabajo reales, se introdujo información epidemiológica real y los resultados obtenidos fueron los siguientes:

| Nº registros | Paquetes enviados | Paquetes recibidos | Bytes totales enviados | Bytes recibidos total |
|---|---|---|---|---|
| **1** | 5 | 5 | 755 | 415 |
| **13** | 65 | 52 | 9877 | 5395 |

| Nº registros | Bytes enviados/registro | Bytes recibidos/registro | Tráfico total/registro | KiB por registro |
|---|---|---|---|---|
| **1** | 755 | 415 | 1170 | 1,14 |
| **13** | 174 | 1000,77 | 1174,77 | 1,14 |



upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

Se puede observar que tanto enviando uno solo como varios a la vez, cada registro, es decir, cada día generaría un tráfico total de unos 1,14 KiB, lo cual resulta más que aceptable.

Según el dispositivo en el que se ha probado la conexión, se obtienen resultados dispares.

Se probó a enviar 16 registros desde un Sony Ericsson K610i vía GPRS con éxito. Después se enviaron otros 16 registros vía UMTS/3G con idéntico resultado.

En cambio, con el emulador de NetBeans, el máximo número de registros que se han conseguido enviar de una sola vez son 13, ya que al intentar enviar el decimocuarto lanza la excepción:

*java.io.IOException: Resource limit exceeded for TCP client sockets*

a pesar de que el socket es cerrado al final de cada envío con la instrucción siguiente:

```
if (hc != null) hc.close();
```

siendo hc un objeto HttpConnection:

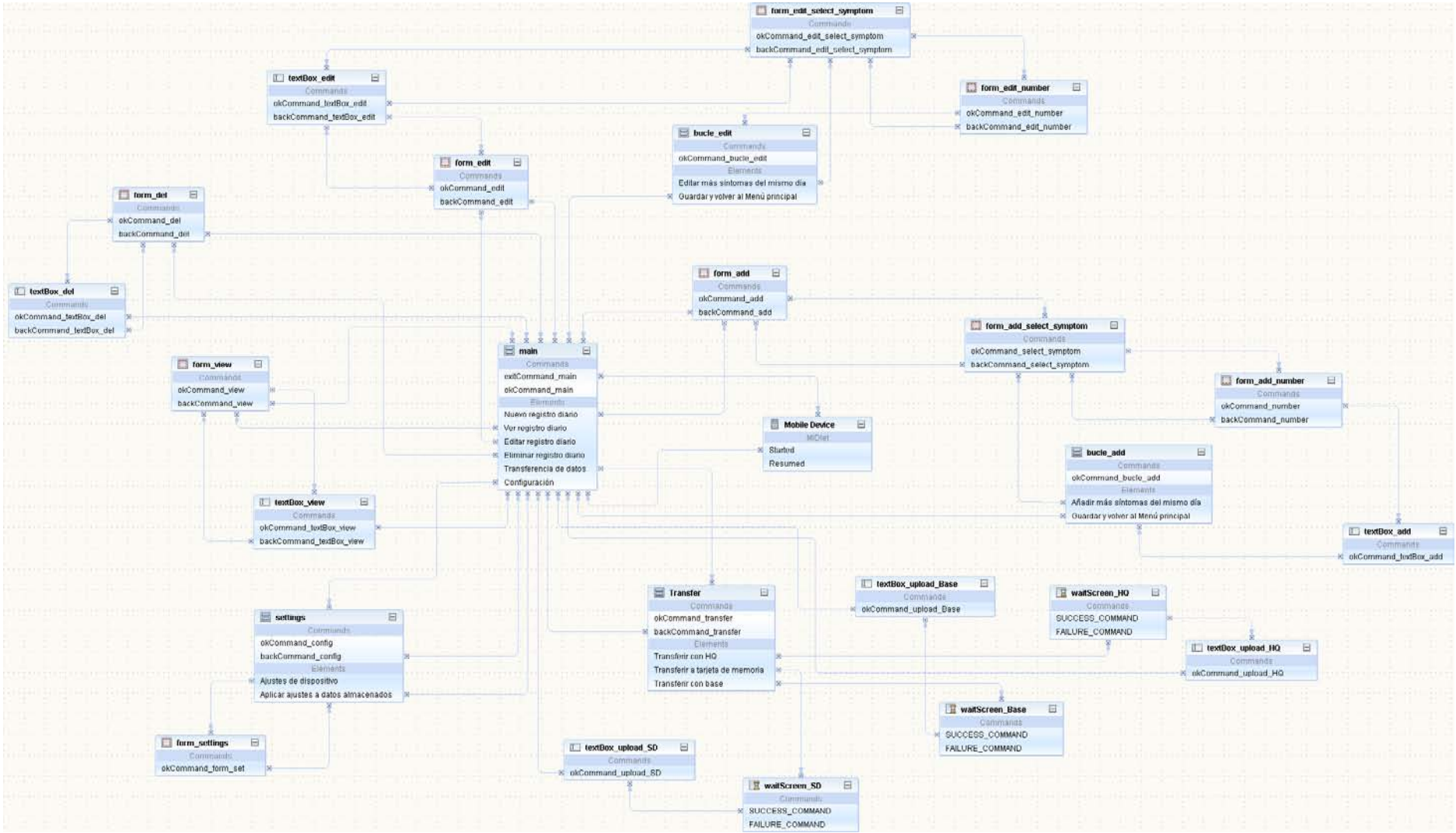
```
hc = (HttpConnection)Connector.open(url, Connector.READ_WRITE);
```

### *Diseño de la interfaz*

A continuación se muestra el flujo de ventanas de la aplicación:



upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira






upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

# Ejemplo de uso

Un ejemplo práctico de utilización de esta aplicación sería en un país en vías de desarrollo devastado por un huracán. El personal médico desplazado atiende a gente que tiene problemas de salud, y va recogiendo datos sobre los síntomas que presentan. Al final del día hacen un recuento de los pacientes, teniendo en cuenta sus afecciones, sexo y rango de edad, e introducen estos datos en la aplicación:

| *12/06/2009* | <5 | | >5 | | SUB-TOTALS | | GRAND |
|---|---|---|---|---|---|---|---|
| | male | female | male | female | male | female | TOTAL |
| **Fever unknown origin** | 0 | 0 | 1 | 0 | 1 | 0 | 1 |
| **acute respiratory infections** | 0 | 2 | 3 | 7 | 3 | 9 | 12 |
| **Malaria (suspected)** | 1 | 0 | 4 | 9 | 5 | 9 | 14 |
| **Malaria (confirm)** | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| **Typhoid fever** | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| **Haemorrhagic Fever** | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| **Jaundice** | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| **Measles** | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| **Watery diarrhoea** | 1 | 3 | 1 | 3 | 2 | 6 | 8 |
| **Bloody diarrhoea** | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| **Tetanus** | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| **Wound/trauma** | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| **Anthrax** | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| **Cutaneous Infection** | 0 | 1 | 4 | 10 | 4 | 11 | 15 |
| **Malnutriton** | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| **Vaginal infection** | | 1 | | 20 | | 21 | 21 |
| **Others** | 0 | 2 | 18 | 23 | 18 | 25 | 43 |



upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

Para introducir estos datos hay que acceder a la primera opción del menú principal: “Nuevo registro diario”.

A continuación, hay que introducir la fecha a través del campo que se ofrece:




upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

Una vez seleccionada la fecha, se muestra una pantalla con los síntomas que permite recoger la aplicación, donde se elegirá uno de ellos.

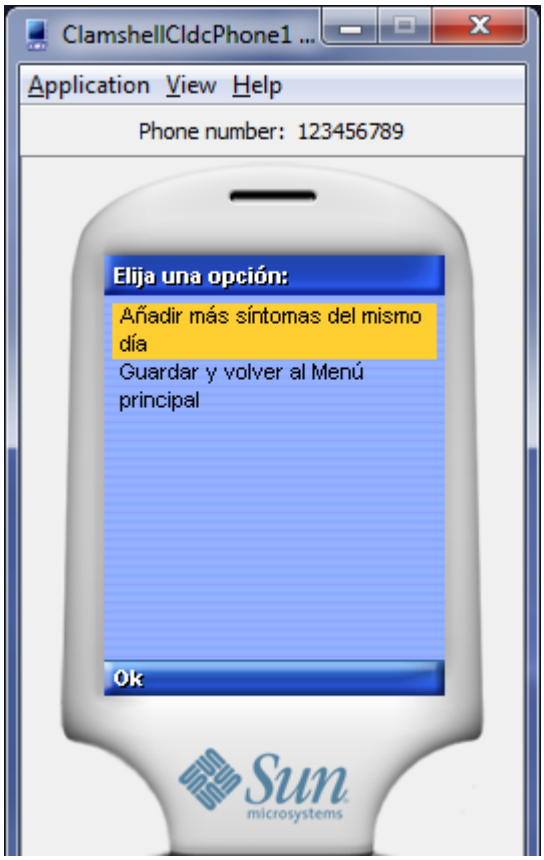
A continuación hay que introducir cuántas personas hay afectadas, clasificadas por sexo y rango de edad.




upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

Para finalizar, hay que elegir entre añadir datos de más síntomas (esta opción puede realizarse todas las veces que haga falta), o finalizar con el registro y guardarlo en disco.

En el caso que se decida añadir más síntomas, volverá a aparecer la pantalla previa de selección de síntoma.






upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

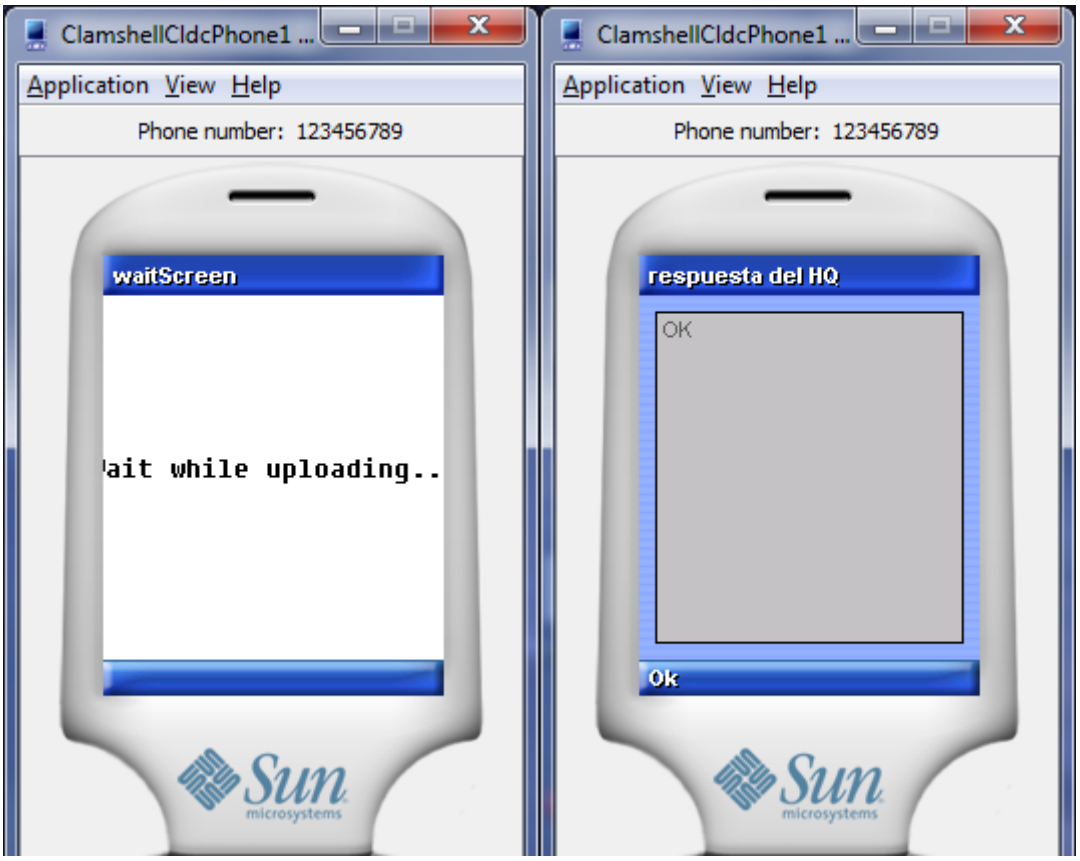
A continuación, se pueden visualizar, editar o eliminar registros ya almacenados a través de sus correspondientes menús.

No obstante, la otra opción significativa de la aplicación consiste en enviar los registros almacenados en el dispositivo. Para ello hay que ir a la opción "Transferencia de datos" y seleccionar Transferir con HQ.




upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

A continuación aparecerá la pantalla de transferencia de datos. Si todo va bien, tras unos segundos aparecerá OK en pantalla tantas veces como registros se hayan transferido correctamente. Los que han llegado bien se mueven a otro fichero de históricos; en cambio, si alguno ha fallado permanece en el fichero de registros pendientes de envío.






upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

# Conclusiones y líneas futuras

## *Conclusiones*

Como se ha podido comprobar, Java Micro Edition resulta una plataforma muy buena para desarrollar aplicaciones destinadas al sector de los dispositivos móviles por ofrecer un entorno multiplataforma.

El problema es que la variedad de dispositivos en este sector es abrumadora, y la portabilidad de la aplicación puede no resultar tan ideal como podría parecer. Aunque las pruebas realizadas en el emulador de NetBeans hayan arrojado resultados óptimos en cuanto a correcto funcionamiento y estabilidad, cuando se procede a instalar el MIDlet en dispositivos reales la estabilidad de la aplicación brilla por su ausencia.

Por ejemplo, la aplicación se instaló en un Sony Ericsson K160i de 4 años que ahora podría ser considerado gama baja-media donde funciona correctamente.

Se instaló también en un Nokia 6630, de gama media-alta aunque algo anticuado, donde funcionaba bien hasta que al editar un registro con datos de diferentes síntomas se cerraba la aplicación.

En cambio, al ponerla en un móvil considerablemente más avanzado como HTC Diamond fallaba al entrar por tercera vez consecutiva a la opción del menú “Ver registro”, que es cuando la aplicación dejaba de responder.

Por tanto se puede observar que la misma compilación para diferentes móviles (y de diferentes fabricantes) no funciona todo lo bien que debería, así que habría que refinarla para cada dispositivo. Como recomendación, creo que lo ideal sería “certificar” una serie de dispositivos con los que se sepa que la aplicación no vaya a dar problemas.

## *Errores detectados*

Como ocurre en todos los ciclos de desarrollos de software, a medida que van avanzando se van detectando errores de codificación y funcionalidad de la aplicación desarrollada. En este caso se han identificado los siguientes fallos, aunque en algún caso podrían considerarse *features*.



upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

### Registros repetidos

No se controla que sólo haya un registro por día, y los requisitos establecían uno diario. En caso de que se deseen añadir más datos de un día concreto, la aplicación ya permite editar registros almacenados. La aplicación considera la fecha de los registros como identificador único, así que si se introduce más de un registro con la misma fecha no habrá consistencia en los datos.

### Añadiendo más síntomas

En la opción del menú para crear un registro nuevo, ésta itera las veces que haga falta para introducir datos sobre más síntomas, pero si en una de estas iteraciones se desea volver atrás, se vuelve a la pantalla de inicio y los datos introducidos hasta el momento en ese registro se pierden.

### Modificación de identificadores

Hay habilitada una opción para modificar los identificadores de operación, base y dispositivo, que muestra correctamente los ajustes pero no permite modificarlos, ya que éstos van en un fichero dentro del .jar, y éste no puede ser modificado durante la ejecución.

### Enfermedades propias de un sexo

Hay síntomas que sólo puede sufrir uno de los dos sexos, como puede ser la infección vaginal. En este caso lo ideal sería que el programa pudiera restringir la introducción de datos sólo al sexo femenino, pero no está implementado, ya que papa ello habría que establecer algún indicador en el fichero editable que contiene los síntomas.

### Almacenamiento en tarjeta de memoria

El MIDlet está configurado para guardar los registros en un fichero dentro de la tarjeta de memoria. El problema es que no todos los dispositivos disponen de esta característica. Lo bueno es que algunos emulan tener tarjeta en una ruta determinada. Habría que sopesar los pros y los contras de mantener esta característica, o de establecer algún mecanismo que permita seleccionar en qué carpeta se desean almacenar.

### Certificación del MIDlet

Otro de los problemas encontrados es que los teléfonos vienen preconfigurados para desconfiar de aplicaciones que no estén firmadas por



upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

entidades de confianza, así que la experiencia de usuario se ve bastante perjudicada debido a las constantes advertencias que lanza la aplicación cada vez que se va a leer o escribir en el sistema de ficheros, o se va a realizar una conexión a Internet.

### *Pruebas de usabilidad*

Se han efectuado pruebas a lo largo del desarrollo en diferentes dispositivos móviles, tanto con teclado físico como con pantalla táctil para estudiar qué tal resulta la experiencia de usuario con este tipo de interfaces móviles.

Pese a que se ha intentado que la interfaz gráfica fuera liviana y no sobrecargara la pantalla con elementos gráficos, tal y como nos aconsejaron expertos, el hecho de tener que introducir una matriz de datos de 4x17 en un aparato tan pequeño resulta algo tedioso.

Si bien los dispositivos que tienen teclado físico ofrecen una experiencia de usuario correcta, mayormente porque hay mayor costumbre de usar este tipo de teclado, las pantallas táctiles resistivas, que necesitan un puntero táctil para introducir datos, pueden resultar bastante incómodas, sobre todo al principio. Una vez que se coge soltura se pueden llegar a introducir datos casi tan rápido como con los teclados físicos, aunque siempre haya un margen de error mayor.

Sería interesante poder implementar la aplicación en terminales con pantalla táctil capacitiva, que funciona sin puntero y por tanto los botones son más cómodos de utilizar por su mayor tamaño y, por qué no, mayor atractivo visual.

### *Líneas futuras*

Una aplicación destinada a la cooperación internacional como ésta, debería soportar la internacionalización, es decir, poder ser traducida a otros idiomas y así lograr una mayor difusión e integración.

También sería adecuado intentar portarla a otras plataformas que poco a poco van entrando en el mercado, como pueden ser Android o iPhone.



upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

Cruz Roja ha propuesto otra forma de introducir datos en la aplicación, ya que la adoptada en este caso puede resultar algo incómoda. Esta proposición supondría introducir los datos sobre el paciente cada vez que éste es atendido. La interfaz consistiría en una pantalla donde se pudiera elegir sexo, rango de edad, y síntoma, y de esta manera el recuento fuera acumulándose a lo largo de la jornada.

También se propone implantar un ordenador en cada base operativa de la operación con conectividad a internet medianamente estable que sea capaz tanto de recoger esta información epidemiológica directamente, como de los dispositivos móviles del personal asociado a dicha base, y enviarla a la base de datos central.



upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

# Bibliografía

## *Libros*

- Java a tope: J2ME (Java 2 Micro Edition). Sergio Gálvez Rojas, Lucas Ortega Díaz. Universidad de Málaga 2003

- Beginning Java ME Platform. Ray Rischpater. Apress 2008

## *Documentación online*

-Documentación de la API de Java ME
http://java.sun.com/javame/reference/apis.jsp

-Documentación de la API de JSON
http://www.json.org/javadoc/index.html

-Documentación de Nokia
http://www.forum.nokia.com/

## *Software de desarrollo*

-Entorno de desarrollo NetBeans IDE 6.8
http://netbeans.org/

## *Proyectos Fin de Carrera*

Estudio de la problemática del tratamiento de datos y de su visualización en situaciones de emergencia e implementación de una solución basada en PDA, PC y servidor. Mikel Belzunegui Goñi, Rafael Huarte Berrueta. Universidad Pública de Navarra 2008.



upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

# Anexo I: Código fuente



upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

```
/**
 * myEpideMobile. Aplicación de Gestión de información epidemiológica en emergencias internacionales.
 *
 * Copyright (C) 2010 Daniel Mayor Azcona daniel.mayor@unavarra.es
 *
 * Este programa es software libre. Puede redistribuirlo y/o modificarlo bajo los términos de la Licencia
 * Pública General de GNU según es publicada por la Free Software Foundation, bien de la versión 2 de dicha
 * Licencia o bien de cualquier versión posterior.
 *
 * Este programa se distribuye con la esperanza de que sea útil, pero SIN NINGUNA GARANTÍA, incluso sin la
 * garantía MERCANTIL implícita o sin garantizar la CONVENIENCIA PARA UN PROPÓSITO PARTICULAR. Véase la Licencia
 * Pública General de GNU para más detalles.
 *
 * Debería haber recibido una copia de la Licencia Pública General junto con este programa. Si no ha sido así,
 * escriba a la Free Software Foundation, Inc., en 675 Mass Ave, Cambridge, MA 02139, EEUU.
 *
 */

import javax.microedition.midlet.MIDlet;
import javax.microedition.lcdui.Alert;
import javax.microedition.lcdui.Choice;
import javax.microedition.lcdui.ChoiceGroup;
import javax.microedition.lcdui.Command;
import javax.microedition.lcdui.CommandListener;
import javax.microedition.lcdui.DateField;
import javax.microedition.lcdui.Display;
import javax.microedition.lcdui.Displayable;
import javax.microedition.lcdui.Form;
import javax.microedition.lcdui.Item;
import javax.microedition.lcdui.List;
import javax.microedition.lcdui.TextField;
import javax.microedition.lcdui.TextBox;
import java.util.Date;
import java.util.Calendar;
import org.json.me.JSONObject;
import org.json.me.JSONException;
import org.json.me.JSONArray;
import java.util.Vector;
import org.netbeans.microedition.lcdui.WaitScreen;
import org.netbeans.microedition.util.SimpleCancellableTask;

/**
 * Contiene el MIDlet de la aplicación con la interfaz gráfica para acceder a las funcionalidades
 * del programa.
 * @author Daniel Mayor
 * @version 1.0
 */
public class menuMIDlet extends MIDlet implements CommandListener {

    private boolean midletPaused = false;
```

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

```
//<editor-fold defaultstate="collapsed" desc=" Generated Fields ">//GEN-BEGIN:|fields|0|
private List main;
private Form form_add;
private DateField dateField_add;
private Form form_view;
private ChoiceGroup choiceGroup_view;
private Form form_edit;
private ChoiceGroup choiceGroup_edit;
private List Transfer;
private List settings;
private Form form_add_select_symptom;
private ChoiceGroup choiceGroup_symptom;
private Form form_add_number;
private TextField textField_mu5;
private TextField textField_fu5;
private TextField textField_mo5;
private TextField textField_fo5;
private TextBox textBox_add;
private List bucle_add;
private TextBox textBox_view;
private Form form_del;
private ChoiceGroup choiceGroup_del;
private TextBox textBox_del;
private Form form_settings;
private TextField ID_Op_field;
private TextField ID_Base_field;
private TextField ID_Device_field;
private List bucle_edit;
private Form form_edit_number;
private TextField textField_ed_fu5;
private TextField textField_ed_fo5;
private TextField textField_ed_mu5;
private TextField textField_ed_mo5;
private Form form_edit_select_symptom;
private ChoiceGroup choiceGroup_edit_symptom;
private TextBox textBox_edit;
private WaitScreen waitScreen_SD;
private WaitScreen waitScreen_Base;
private TextBox textBox_upload_Base;
private TextBox textBox_upload_SD;
private WaitScreen waitScreen_HQ;
private TextBox textBox_upload_HQ;
private Command exitCommand_main;
private Command okCommand_add;
private Command okCommand_main;
private Command backCommand_add;
private Command okCommand_edit;
private Command backCommand_view;
private Command backCommand_edit;
private Command backCommand_transfer;
private Command backCommand_config;
private Command backCommand_select_symptom;
private Command okCommand_number;
private Command okCommand_select_symptom;
private Command backCommand_number;
private Command okCommand_textBox_view;
```

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

```
    private Command okCommand_view;
    private Command okCommand_textBox_add;
    private Command okCommand_bucle_add;
    private Command okCommand_del;
    private Command backCommand_del;
    private Command okCommand_config;
    private Command backCommand_textBox_view;
    private Command okCommand_textBox_del;
    private Command backCommand_textBox_del;
    private Command okCommand_form_set;
    private Command okCommand_bucle_edit;
    private Command backCommand_edit_number;
    private Command okCommand_edit_number;
    private Command backCommand_edit_select_symptom;
    private Command okCommand_edit_select_symptom;
    private Command backCommand_textBox_edit;
    private Command okCommand_textBox_edit;
    private Command okCommand_upload_SD;
    private Command okCommand_upload_Base;
    private Command okCommand_transfer;
    private Command okCommand_upload_HQ;
    private SimpleCancellableTask task_edit;
    private SimpleCancellableTask task_upload_SD;
    private SimpleCancellableTask task_upload_Base;
    private SimpleCancellableTask task_upload_HQ;
    //</editor-fold>//GEN-END:|fields|0|
    String id_operation;
    String id_base;
    String id_device;
    String date_add, date_view, date_edit, date_del;
    String veamos = null;
    String stringjson;
    String str_view;            //string con el que se muestra el contenido del registro a
visualizar
    String str_edit;            //string con el que se muestra el contenido del registro a
editar
    String str_del;             //string con el que se muestrael contenido del registro a
borrar
    String urlserver;
    String urlquery;
    String f_rec;               //nombre del fichero de registros
    String ruta;
    String[] return_HQ, return_SD;
    String[] ids;               //array donde se almacenan los ids de la op, base y dev
    String[] diseases;          //array donde se guardan los nombres de los síntomas
    String[] jsons;             //array donde se guardan los json guardados en disco sin
enviar
    String[][] data;            //array de 2 dimensiones utilizado para guardar por separado
el numero de pacientes
    int disease;
    int exito;
    int number_of_diseases;     //número de síntomas
    int n_recs;                 //número de registros almacenados en fichero de texto
    int suc;                    //resultado de la operación de editar un registro
    boolean[] booleano, bool_view, bool_del;//array de booleanos utilizado para generar el
formulario select_diseases
    JSONObject[] jObject;       //array de objetos json
```

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

```
    JSONObject json_view;       //json donde se almacena el registro que se desea visualizar
    JSONObject json_edit;       //json donde se almacena el registro que se desea editar
    JSONObject json_del;        //json donde se almacena el registro que se desea eliminar
    JSONArray arr_edit_mu5;     //arrays donde se meten los arrays de síntomas organizados
por sexo y edad
    JSONArray arr_edit_mo5;
    JSONArray arr_edit_fu5;
    JSONArray arr_edit_fo5;
    Vector vjson_view = null;   //Vector donde se mete el contenido de jsons a visualizar
    Vector vjson_edit = null;   //Vector donde se mete el contenido de jsons a editar
    Vector vjson_del = null;    //Vector donde se mete el contenido de jsons a eliminar
    getConfigSymptom symptoms;  //objeto con el que se accede a los síntomas almacenados en
.txt
    getConfigIDs conf;          //objeto con el que se accede a los conf almacenados en .txt
    getJSON gjson_view;         //objeto con el que se accede a los json almacenados en .txt
para visualizar (provisional)

    /**
     * The menuMIDlet MIDlet constructor.
     */
    public menuMIDlet() {
        conf = new getConfigIDs("settings.txt");//obtenemos configuración
        ids = conf.idstring();
        symptoms = new getConfigSymptom("symptoms.txt");//accedemos al fichero de síntomas
        number_of_diseases = symptoms.n_sintomas();     //Obtenemos número de síntomas
        diseases = symptoms.sintomas();                 //Obtenemos síntomas
        id_operation = ids[0];
        id_base = ids[1];
        id_device = ids[2];
        urlserver = ids[3];
        urlquery = ids[4];
        f_rec = ids[5];
        ruta = ids[6];
    }

    //<editor-fold defaultstate="collapsed" desc=" Generated Methods ">//GEN-BEGIN:|methods|0|
    //</editor-fold>//GEN-END:|methods|0|

    //<editor-fold defaultstate="collapsed" desc=" Generated Method: initialize
">//GEN-BEGIN:|0-initialize|0|0-preInitialize
    /**
     * Initilizes the application.
     * It is called only once when the MIDlet is started. The method is called before the
<code>startMIDlet</code> method.
     */
    private void initialize() {//GEN-END:|0-initialize|0|0-preInitialize
        // write pre-initialize user code here
//GEN-LINE:|0-initialize|1|0-postInitialize
        // write post-initialize user code here
    }//GEN-BEGIN:|0-initialize|2|
    //</editor-fold>//GEN-END:|0-initialize|2|

    //<editor-fold defaultstate="collapsed" desc=" Generated Method: startMIDlet
">//GEN-BEGIN:|3-startMIDlet|0|3-preAction
    /**
     * Performs an action assigned to the Mobile Device - MIDlet Started point.
```

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

```
    public void startMIDlet() {//GEN-END:|3-startMIDlet|0|3-preAction
        // write pre-action user code here
        switchDisplayable(null, getMain());//GEN-LINE:|3-startMIDlet|1|3-postAction
        // write post-action user code here
    }//GEN-BEGIN:|3-startMIDlet|2|
    //</editor-fold>//GEN-END:|3-startMIDlet|2|

    //<editor-fold defaultstate="collapsed" desc=" Generated Method: resumeMIDlet
">//GEN-BEGIN:|4-resumeMIDlet|0|4-preAction
    /**
     * Performs an action assigned to the Mobile Device - MIDlet Resumed point.
     */
    public void resumeMIDlet() {//GEN-END:|4-resumeMIDlet|0|4-preAction
        // write pre-action user code here
//GEN-LINE:|4-resumeMIDlet|1|4-postAction
        // write post-action user code here
    }//GEN-BEGIN:|4-resumeMIDlet|2|
    //</editor-fold>//GEN-END:|4-resumeMIDlet|2|

    //<editor-fold defaultstate="collapsed" desc=" Generated Method: switchDisplayable
">//GEN-BEGIN:|5-switchDisplayable|0|5-preSwitch
    /**
     * Switches a current displayable in a display. The <code>display</code> instance is
taken from <code>getDisplay</code> method. This method is used by all actions in the design
for switching displayable.
     * @param alert the Alert which is temporarily set to the display; if <code>null</code>,
then <code>nextDisplayable</code> is set immediately
     * @param nextDisplayable the Displayable to be set
     */
    public void switchDisplayable(Alert alert, Displayable nextDisplayable) {
//GEN-END:|5-switchDisplayable|0|5-preSwitch
        // write pre-switch user code here
        Display display = getDisplay();//GEN-BEGIN:|5-switchDisplayable|1|5-postSwitch
        if (alert == null) {
            display.setCurrent(nextDisplayable);
        } else {
            display.setCurrent(alert, nextDisplayable);
        }//GEN-END:|5-switchDisplayable|1|5-postSwitch
        // write post-switch user code here
    }//GEN-BEGIN:|5-switchDisplayable|2|
    //</editor-fold>//GEN-END:|5-switchDisplayable|2|

    //<editor-fold defaultstate="collapsed" desc=" Generated Method: commandAction for
Displayables ">//GEN-BEGIN:|7-commandAction|0|7-preCommandAction
    /**
     * Called by a system to indicated that a command has been invoked on a particular
displayable.
     * @param command the Command that was invoked
     * @param displayable the Displayable where the command was invoked
     */
    public void commandAction(Command command, Displayable displayable) {
//GEN-END:|7-commandAction|0|7-preCommandAction
        // write pre-action user code here
        if (displayable == Transfer) {//GEN-BEGIN:|7-commandAction|1|108-preAction
            if (command == List.SELECT_COMMAND) {//GEN-END:|7-commandAction|1|108-preAction
                // write pre-action user code here
                TransferAction();//GEN-LINE:|7-commandAction|2|108-postAction
```

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

```
                // write post-action user code here
            } else if (command == backCommand_transfer) {
//GEN-LINE:|7-commandAction|3|113-preAction
                // write pre-action user code here
                switchDisplayable(null, getMain());
//GEN-LINE:|7-commandAction|4|113-postAction
                // write post-action user code here
            } else if (command == okCommand_transfer) {
//GEN-LINE:|7-commandAction|5|490-preAction
                // write pre-action user code here
                if (Transfer.getSelectedIndex()==0) {           //transferir a HQ
                    switchDisplayable(null, getWaitScreen_HQ());
                }
                else if (Transfer.getSelectedIndex()==1) {      //transferir a tarjeta de
memoria
                    //switchDisplayable(null, getWaitScreen_SD());
                }
                else if (Transfer.getSelectedIndex()==2) {      //transferir a base
                    //switchDisplayable(null, getWaitScreen_Base());
                }
//GEN-LINE:|7-commandAction|6|490-postAction
                // write post-action user code here
            }//GEN-BEGIN:|7-commandAction|7|369-preAction
        } else if (displayable == bucle_add) {
            if (command == List.SELECT_COMMAND) {//GEN-END:|7-commandAction|7|369-preAction
                // write pre-action user code here
//GEN-LINE:|7-commandAction|8|369-postAction
                // write post-action user code here
            } else if (command == okCommand_bucle_add) {
//GEN-LINE:|7-commandAction|9|371-preAction
                // write pre-action user code here
                if (bucle_add.getSelectedIndex()==0) {  //seleccionamos agregar más datos al
registro
                    resetForm_add();
                    if (choiceGroup_symptom != null)
                        choiceGroup_symptom.setSelectedIndex(disease+1, true);
                    switchDisplayable(null, getForm_add_select_symptom());
                }
                else if (bucle_add.getSelectedIndex()==1) { //seleccionamos guardar registro
y volver al Menú
                    stringjson = generaJSON();
                        saveData sd = new saveData();
                        exito = sd.saveData(f_rec, stringjson, false);
                    data = null;
                    vjson_del = vjson_edit = vjson_view = null;
                    choiceGroup_del = choiceGroup_edit = choiceGroup_view = null;
                    switchDisplayable(null, getMain());
                }
//GEN-LINE:|7-commandAction|10|371-postAction
                // write post-action user code here
            }//GEN-BEGIN:|7-commandAction|11|475-preAction
        } else if (displayable == bucle_edit) {
            if (command == List.SELECT_COMMAND) {//GEN-END:|7-commandAction|11|475-preAction
                // write pre-action user code here
                bucle_editAction();//GEN-LINE:|7-commandAction|12|475-postAction
                // write post-action user code here
            } else if (command == okCommand_bucle_edit) {
```

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

```
//GEN-LINE:|7-commandAction|13|479-preAction
                // write pre-action user code here
                if (bucle_edit.getSelectedIndex()==0) {  //opción agregar más datos al
registro
                    resetForm_edit();//reseteo de algunos elementos del formulario
                    switchDisplayable(null, getForm_edit_select_symptom());
                }
                else if (bucle_edit.getSelectedIndex()==1) { //opción guardar registro y
volver al Menú
                    saveData sd_edit = new saveData();
                    sd_edit.saveData(f_rec, jObject, true);
                    sd_edit = null;
                    vjson_del = vjson_edit = vjson_view = null;
                    choiceGroup_del = choiceGroup_edit = choiceGroup_view = null;
                    form_del = form_edit = form_view = null;
                    switchDisplayable(null, getMain());
                }
//GEN-LINE:|7-commandAction|14|479-postAction
                // write post-action user code here
            }//GEN-BEGIN:|7-commandAction|15|77-preAction
        } else if (displayable == form_add) {
            if (command == backCommand_add) {//GEN-END:|7-commandAction|15|77-preAction
                // write pre-action user code here
                switchDisplayable(null, getMain());
//GEN-LINE:|7-commandAction|16|77-postAction
                // write post-action user code here
            } else if (command == okCommand_add) {//GEN-LINE:|7-commandAction|17|71-preAction
                // write pre-action user code here
                switchDisplayable(null, getForm_add_select_symptom());
//GEN-LINE:|7-commandAction|18|71-postAction
                // write post-action user code here
            }//GEN-BEGIN:|7-commandAction|19|334-preAction
        } else if (displayable == form_add_number) {
            if (command == backCommand_number) {//GEN-END:|7-commandAction|19|334-preAction
                // write pre-action user code here
                switchDisplayable(null, getForm_add_select_symptom());
//GEN-LINE:|7-commandAction|20|334-postAction
                // write post-action user code here
            } else if (command == okCommand_number) {
//GEN-LINE:|7-commandAction|21|256-preAction
                // write pre-action user code here
                switchDisplayable(null, getTextBox_add());
//GEN-LINE:|7-commandAction|22|256-postAction
                // write post-action user code here

            }//GEN-BEGIN:|7-commandAction|23|260-preAction
        } else if (displayable == form_add_select_symptom) {
            if (command == backCommand_select_symptom) {
//GEN-END:|7-commandAction|23|260-preAction
                // write pre-action user code here
                switchDisplayable(null, getForm_add());
//GEN-LINE:|7-commandAction|24|260-postAction
                // write post-action user code here
            } else if (command == okCommand_select_symptom) {
//GEN-LINE:|7-commandAction|25|292-preAction
                // write pre-action user code here
                switchDisplayable(null, getForm_add_number());
```

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

```
//GEN-LINE:|7-commandAction|26|292-postAction
                // write post-action user code here
            }//GEN-BEGIN:|7-commandAction|27|410-preAction
        } else if (displayable == form_del) {
            if (command == backCommand_del) {//GEN-END:|7-commandAction|27|410-preAction
                // write pre-action user code here
                switchDisplayable(null, getMain());
//GEN-LINE:|7-commandAction|28|410-postAction
                // write post-action user code here
            } else if (command == okCommand_del) {//GEN-LINE:|7-commandAction|29|412-preAction
                // write pre-action user code here
                switchDisplayable(null, getTextBox_del());
//GEN-LINE:|7-commandAction|30|412-postAction
                // write post-action user code here
                if (vjson_del.isEmpty())
                    textBox_del.setString("vjson_del vacío");
                else if (choiceGroup_del == null)
                    textBox_del.setString("choiceGroup null");
            }//GEN-BEGIN:|7-commandAction|31|81-preAction
        } else if (displayable == form_edit) {
            if (command == backCommand_edit) {//GEN-END:|7-commandAction|31|81-preAction
                // write pre-action user code here
                switchDisplayable(null, getMain());
//GEN-LINE:|7-commandAction|32|81-postAction
                // write post-action user code here
            } else if (command == okCommand_edit) {//GEN-LINE:|7-commandAction|33|79-preAction
                // write pre-action user code here
                switchDisplayable(null, getTextBox_edit());
//GEN-LINE:|7-commandAction|34|79-postAction
                // write post-action user code here
            }//GEN-BEGIN:|7-commandAction|35|473-preAction
        } else if (displayable == form_edit_number) {
            if (command == backCommand_edit_number) {
//GEN-END:|7-commandAction|35|473-preAction
                // write pre-action user code here
                textField_ed_mu5.setString("");
                textField_ed_mo5.setString("");
                textField_ed_fu5.setString("");
                textField_ed_fo5.setString("");
                switchDisplayable(null, getForm_edit_select_symptom());
//GEN-LINE:|7-commandAction|36|473-postAction
                // write post-action user code here
            } else if (command == okCommand_edit_number) {
//GEN-LINE:|7-commandAction|37|471-preAction
                // write pre-action user code here
                switchDisplayable(null, getBucle_edit());
//GEN-LINE:|7-commandAction|38|471-postAction
                // write post-action user code here
            }//GEN-BEGIN:|7-commandAction|39|461-preAction
        } else if (displayable == form_edit_select_symptom) {
            if (command == backCommand_edit_select_symptom) {
//GEN-END:|7-commandAction|39|461-preAction
                // write pre-action user code here
                switchDisplayable(null, getTextBox_edit());
//GEN-LINE:|7-commandAction|40|461-postAction
                // write post-action user code here
            } else if (command == okCommand_edit_select_symptom) {
```

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

```
//GEN-LINE:|7-commandAction|41|459-preAction
                // write pre-action user code here
                switchDisplayable(null, getForm_edit_number());
//GEN-LINE:|7-commandAction|42|459-postAction
                // write post-action user code here
            }//GEN-BEGIN:|7-commandAction|43|432-preAction
        } else if (displayable == form_settings) {
            if (command == okCommand_form_set) {//GEN-END:|7-commandAction|43|432-preAction
                // write pre-action user code here
                switchDisplayable(null, getSettings());
//GEN-LINE:|7-commandAction|44|432-postAction
                // write post-action user code here
                /*
                ids[0] = ID_Op_field.getString();
                ids[1] = ID_Base_field.getString();
                ids[2] = ID_Device_field.getString();*/
                /*try {
                    ids.put("operation", ID_Op_field.getString());
                    ids.put("base", ID_Base_field.getString());
                    ids.put("device", ID_Device_field.getString());
                } catch (JSONException jsone) { jsone.printStackTrace(); }
                saveData sd_set = new saveData();*/
                //exito = sd_set.saveData("ids.txt", ids[0]+"\n"+ids[1]+"\n"+ids[2]);
            }//GEN-BEGIN:|7-commandAction|45|75-preAction
        } else if (displayable == form_view) {
            if (command == backCommand_view) {//GEN-END:|7-commandAction|45|75-preAction
                // write pre-action user code here
                switchDisplayable(null, getMain());
//GEN-LINE:|7-commandAction|46|75-postAction
                // write post-action user code here
            } else if (command == okCommand_view) {
//GEN-LINE:|7-commandAction|47|378-preAction
                // write pre-action user code here
                switchDisplayable(null, getTextBox_view());
//GEN-LINE:|7-commandAction|48|378-postAction
                // write post-action user code here
            }//GEN-BEGIN:|7-commandAction|49|26-preAction
        } else if (displayable == main) {
            if (command == List.SELECT_COMMAND) {//GEN-END:|7-commandAction|49|26-preAction
                // write pre-action user code here
                mainAction();//GEN-LINE:|7-commandAction|50|26-postAction
                // write post-action user code here
            } else if (command == exitCommand_main) {
//GEN-LINE:|7-commandAction|51|34-preAction
                // write pre-action user code here
                exitMIDlet();//GEN-LINE:|7-commandAction|52|34-postAction
                // write post-action user code here
            } else if (command == okCommand_main) {
//GEN-LINE:|7-commandAction|53|313-preAction
                // write pre-action user code here
                if (main.getSelectedIndex()==0) {
                    resetForm_add();    //reseteamos formulario por si no se ha reseteado
antes
                    switchDisplayable(null, getForm_add());
                }
                else if (main.getSelectedIndex()==1) {
                    switchDisplayable(null, getForm_view());
```

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

```
                }
                else if (main.getSelectedIndex()==2) {
                    resetForm_edit();
                    if (choiceGroup_edit != null) {
                        choiceGroup_edit = null;
                    }
                    switchDisplayable(null, getForm_edit());
                }
                else if (main.getSelectedIndex()==3)
                    switchDisplayable(null, getForm_del());
                else if (main.getSelectedIndex()==4)
                    switchDisplayable(null, getTransfer());
                else if (main.getSelectedIndex()==5)
                    switchDisplayable(null, getSettings());
//GEN-LINE:|7-commandAction|54|313-postAction
                // write post-action user code here
                main.setSelectedIndex(0, true);
            }//GEN-BEGIN:|7-commandAction|55|141-preAction
        } else if (displayable == settings) {
            if (command == List.SELECT_COMMAND) {//GEN-END:|7-commandAction|55|141-preAction
                // write pre-action user code here
                settingsAction();//GEN-LINE:|7-commandAction|56|141-postAction
                // write post-action user code here
            } else if (command == backCommand_config) {
//GEN-LINE:|7-commandAction|57|171-preAction
                // write pre-action user code here
                switchDisplayable(null, getMain());
//GEN-LINE:|7-commandAction|58|171-postAction
                // write post-action user code here
            } else if (command == okCommand_config) {
//GEN-LINE:|7-commandAction|59|388-preAction
                // write pre-action user code here
                if (settings.getSelectedIndex()==0)
                    switchDisplayable(null, getForm_settings());
                else if (settings.getSelectedIndex()==1)
                    //pero antes hay que modificar todos los registros del fichero...
                    switchDisplayable(null, getMain());
//GEN-LINE:|7-commandAction|60|388-postAction
                // write post-action user code here
            }//GEN-BEGIN:|7-commandAction|61|360-preAction
        } else if (displayable == textBox_add) {
            if (command == okCommand_textBox_add) {//GEN-END:|7-commandAction|61|360-preAction
                // write pre-action user code here
                switchDisplayable(null, getBucle_add());
//GEN-LINE:|7-commandAction|62|360-postAction
                // write post-action user code here
            }//GEN-BEGIN:|7-commandAction|63|446-preAction
        } else if (displayable == textBox_del) {
            if (command == backCommand_textBox_del) {
//GEN-END:|7-commandAction|63|446-preAction
                // write pre-action user code here
                //si se retrocede, se resetean estas variables para poder seleccionar otro
registro
                str_del = null;
                textBox_del.setString("");
                textBox_del = null;
                switchDisplayable(null, getForm_del());
```

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

```
//GEN-LINE:|7-commandAction|64|446-postAction
                // write post-action user code here
            } else if (command == okCommand_textBox_del) {
//GEN-LINE:|7-commandAction|65|442-preAction
                // write pre-action user code here
                try {//se borra el elemento seleccionado para borrar, y si no hay ninguno se
vuelve al menú
                    vjson_del.removeElementAt(choiceGroup_del.getSelectedIndex());
                    vjson_edit = vjson_view = vjson_del;
                    saveData sd_del = new saveData();                //declaramos el objeto
saveData
                    sd_del.saveData(f_rec, vjson_del, true);         //sobrescribimos
eliminados.txt
                }
                catch (java.lang.ArrayIndexOutOfBoundsException e) {
                    textBox_del.setString(e.toString());
                }
                catch (Exception e) {
                    textBox_del.setString(e.toString());
                }
                resetForm_del();
                choiceGroup_del = choiceGroup_edit = choiceGroup_view = null;
                switchDisplayable(null, getMain());
//GEN-LINE:|7-commandAction|66|442-postAction
                // write post-action user code here
            }//GEN-BEGIN:|7-commandAction|67|454-preAction
        } else if (displayable == textBox_edit) {
            if (command == backCommand_textBox_edit) {
//GEN-END:|7-commandAction|67|454-preAction
                // write pre-action user code here
                textBox_edit = null;
                switchDisplayable(null, getForm_edit());
//GEN-LINE:|7-commandAction|68|454-postAction
                // write post-action user code here
            } else if (command == okCommand_textBox_edit) {
//GEN-LINE:|7-commandAction|69|452-preAction
                // write pre-action user code here
                switchDisplayable(null, getForm_edit_select_symptom());
//GEN-LINE:|7-commandAction|70|452-postAction
                // write post-action user code here
            }//GEN-BEGIN:|7-commandAction|71|499-preAction
        } else if (displayable == textBox_upload_Base) {
            if (command == okCommand_upload_Base) {//GEN-END:|7-commandAction|71|499-preAction
                // write pre-action user code here
                switchDisplayable(null, getMain());
//GEN-LINE:|7-commandAction|72|499-postAction
                // write post-action user code here
            }//GEN-BEGIN:|7-commandAction|73|487-preAction
        } else if (displayable == textBox_upload_HQ) {
            if (command == okCommand_upload_HQ) {//GEN-END:|7-commandAction|73|487-preAction
                // write pre-action user code here
                switchDisplayable(null, getMain());
//GEN-LINE:|7-commandAction|74|487-postAction
                // write post-action user code here
            }//GEN-BEGIN:|7-commandAction|75|501-preAction
        } else if (displayable == textBox_upload_SD) {
            if (command == okCommand_upload_SD) {//GEN-END:|7-commandAction|75|501-preAction
```

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

```
                // write pre-action user code here
                switchDisplayable(null, getMain());
//GEN-LINE:|7-commandAction|76|501-postAction
                // write post-action user code here
            }//GEN-BEGIN:|7-commandAction|77|385-preAction
        } else if (displayable == textBox_view) {
            if (command == backCommand_textBox_view) {
//GEN-END:|7-commandAction|77|385-preAction
                // write pre-action user code here
                switchDisplayable(null, getForm_view());
//GEN-LINE:|7-commandAction|78|385-postAction
                // write post-action user code here
                textBox_view = null;
                str_view = "";
            } else if (command == okCommand_textBox_view) {
//GEN-LINE:|7-commandAction|79|382-preAction
                // write pre-action user code here
                switchDisplayable(null, getMain());
//GEN-LINE:|7-commandAction|80|382-postAction
                // write post-action user code here
                textBox_view = null;
                str_view = "";
            }//GEN-BEGIN:|7-commandAction|81|506-preAction
        } else if (displayable == waitScreen_Base) {
            if (command == WaitScreen.FAILURE_COMMAND) {
//GEN-END:|7-commandAction|81|506-preAction
                // write pre-action user code here
//GEN-LINE:|7-commandAction|82|506-postAction
                // write post-action user code here
            } else if (command == WaitScreen.SUCCESS_COMMAND) {
//GEN-LINE:|7-commandAction|83|505-preAction
                // write pre-action user code here
                switchDisplayable(null, getTextBox_upload_Base());
//GEN-LINE:|7-commandAction|84|505-postAction
                // write post-action user code here
            }//GEN-BEGIN:|7-commandAction|85|493-preAction
        } else if (displayable == waitScreen_HQ) {
            if (command == WaitScreen.FAILURE_COMMAND) {
//GEN-END:|7-commandAction|85|493-preAction
                // write pre-action user code here
//GEN-LINE:|7-commandAction|86|493-postAction
                // write post-action user code here
            } else if (command == WaitScreen.SUCCESS_COMMAND) {
//GEN-LINE:|7-commandAction|87|492-preAction
                // write pre-action user code here
                switchDisplayable(null, getTextBox_upload_HQ());
//GEN-LINE:|7-commandAction|88|492-postAction
                // write post-action user code here
            }//GEN-BEGIN:|7-commandAction|89|510-preAction
        } else if (displayable == waitScreen_SD) {
            if (command == WaitScreen.FAILURE_COMMAND) {
//GEN-END:|7-commandAction|89|510-preAction
                // write pre-action user code here
//GEN-LINE:|7-commandAction|90|510-postAction
                // write post-action user code here
            } else if (command == WaitScreen.SUCCESS_COMMAND) {
//GEN-LINE:|7-commandAction|91|509-preAction
```

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

```
                // write pre-action user code here
                switchDisplayable(null, getTextBox_upload_SD());
//GEN-LINE:|7-commandAction|92|509-postAction
                // write post-action user code here
            }//GEN-BEGIN:|7-commandAction|93|7-postCommandAction
        }//GEN-END:|7-commandAction|93|7-postCommandAction
        // write post-action user code here
    }//GEN-BEGIN:|7-commandAction|94|
    //</editor-fold>//GEN-END:|7-commandAction|94|


    //<editor-fold defaultstate="collapsed" desc=" Generated Getter: main
">//GEN-BEGIN:|25-getter|0|25-preInit
    /**
     * Returns an initiliazed instance of main component.
     * @return the initialized component instance
     */
    public List getMain() {
        if (main == null) {//GEN-END:|25-getter|0|25-preInit
            // write pre-init user code here
            main = new List("Men\u00FA", Choice.IMPLICIT);//GEN-BEGIN:|25-getter|1|25-postInit
            main.append("Nuevo registro diario", null);
            main.append("Ver registro diario", null);
            main.append("Editar registro diario", null);
            main.append("Eliminar registro diario", null);
            main.append("Transferencia de datos", null);
            main.append("Configuraci\u00F3n", null);
            main.addCommand(getExitCommand_main());
            main.addCommand(getOkCommand_main());
            main.setCommandListener(this);
            main.setFitPolicy(Choice.TEXT_WRAP_DEFAULT);
            main.setSelectedFlags(new boolean[] { false, false, false, false, false, false
});//GEN-END:|25-getter|1|25-postInit
            // write post-init user code here
        }//GEN-BEGIN:|25-getter|2|
        return main;
    }
    //</editor-fold>//GEN-END:|25-getter|2|

    //<editor-fold defaultstate="collapsed" desc=" Generated Method: mainAction
">//GEN-BEGIN:|25-action|0|25-preAction
    /**
     * Performs an action assigned to the selected list element in the main component.
     */
    public void mainAction() {//GEN-END:|25-action|0|25-preAction
        // enter pre-action user code here
        String __selectedString = getMain().getString(getMain().getSelectedIndex());
//GEN-BEGIN:|25-action|1|29-preAction
        if (__selectedString != null) {
            if (__selectedString.equals("Nuevo registro diario")) {
//GEN-END:|25-action|1|29-preAction
                // write pre-action user code here
                resetForm_add();    //reseteamos formulario por si no se ha reseteado antes
                switchDisplayable(null, getForm_add());//GEN-LINE:|25-action|2|29-postAction
                // write post-action user code here
            } else if (__selectedString.equals("Ver registro diario")) {
//GEN-LINE:|25-action|3|41-preAction
```

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

```
                // write pre-action user code here
                if (choiceGroup_view != null) {
                    choiceGroup_view = null;
                }
                switchDisplayable(null, getForm_view());//GEN-LINE:|25-action|4|41-postAction
                // write post-action user code here
            } else if (__selectedString.equals("Editar registro diario")) {
//GEN-LINE:|25-action|5|30-preAction
                // write pre-action user code here
                resetForm_edit();
                if (choiceGroup_edit != null) {
                    choiceGroup_edit = null;
                }
                switchDisplayable(null, getForm_edit());//GEN-LINE:|25-action|6|30-postAction
                // write post-action user code here
            } else if (__selectedString.equals("Eliminar registro diario")) {
//GEN-LINE:|25-action|7|31-preAction
                // write pre-action user code here
                switchDisplayable(null, getForm_del());//GEN-LINE:|25-action|8|31-postAction
                // write post-action user code here
            } else if (__selectedString.equals("Transferencia de datos")) {
//GEN-LINE:|25-action|9|32-preAction
                // write pre-action user code here
                switchDisplayable(null, getTransfer());//GEN-LINE:|25-action|10|32-postAction
                // write post-action user code here
            } else if (__selectedString.equals("Configuraci\u00F3n")) {
//GEN-LINE:|25-action|11|118-preAction
                // write pre-action user code here
                switchDisplayable(null, getSettings());//GEN-LINE:|25-action|12|118-postAction
                // write post-action user code here
            }//GEN-BEGIN:|25-action|13|25-postAction
        }//GEN-END:|25-action|13|25-postAction
        // enter post-action user code here
    }//GEN-BEGIN:|25-action|14|
    //</editor-fold>//GEN-END:|25-action|14|

    //<editor-fold defaultstate="collapsed" desc=" Generated Getter: exitCommand_main
">//GEN-BEGIN:|33-getter|0|33-preInit
    /**
     * Returns an initiliazed instance of exitCommand_main component.
     * @return the initialized component instance
     */
    public Command getExitCommand_main() {
        if (exitCommand_main == null) {//GEN-END:|33-getter|0|33-preInit
            // write pre-init user code here
            exitCommand_main = new Command("Exit", Command.EXIT, 0);
//GEN-LINE:|33-getter|1|33-postInit
            // write post-init user code here
        }//GEN-BEGIN:|33-getter|2|
        return exitCommand_main;
    }
    //</editor-fold>//GEN-END:|33-getter|2|

    //<editor-fold defaultstate="collapsed" desc=" Generated Getter: form_add
">//GEN-BEGIN:|45-getter|0|45-preInit
    /**
     * Returns an initiliazed instance of form_add component.
```

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

```
     * @return the initialized component instance
     */
    public Form getForm_add() {
        if (form_add == null) {//GEN-END:|45-getter|0|45-preInit
            // write pre-init user code here
            //inicializamos registro diario a 0
            data = new String[4][number_of_diseases];
            for (int i = 0; i < 4; i++)
                for (int j = 0; j < number_of_diseases; j++)
                        data[i][j] = "0";
            form_add = new Form("Nuevo registro diario", new Item[] { getDateField_add() });
//GEN-BEGIN:|45-getter|1|45-postInit
            form_add.addCommand(getOkCommand_add());
            form_add.addCommand(getBackCommand_add());
            form_add.setCommandListener(this);//GEN-END:|45-getter|1|45-postInit
            // write post-init user code here
        }//GEN-BEGIN:|45-getter|2|
        return form_add;
    }
    //</editor-fold>//GEN-END:|45-getter|2|

    //<editor-fold defaultstate="collapsed" desc=" Generated Getter: dateField_add
">//GEN-BEGIN:|47-getter|0|47-preInit
    /**
     * Returns an initiliazed instance of dateField_add component.
     * @return the initialized component instance
     */
    public DateField getDateField_add() {
        if (dateField_add == null) {//GEN-END:|47-getter|0|47-preInit
            // write pre-init user code here
            dateField_add = new DateField("Elegir fecha", DateField.DATE);
//GEN-BEGIN:|47-getter|1|47-postInit
            dateField_add.setDate(new java.util.Date(System.currentTimeMillis()));
//GEN-END:|47-getter|1|47-postInit
            // write post-init user code here
        }//GEN-BEGIN:|47-getter|2|
        return dateField_add;
    }
    //</editor-fold>//GEN-END:|47-getter|2|

    //<editor-fold defaultstate="collapsed" desc=" Generated Getter: form_view
">//GEN-BEGIN:|56-getter|0|56-preInit
    /**
     * Returns an initiliazed instance of form_view component.
     * @return the initialized component instance
     */
    public Form getForm_view() {
        if (form_view == null) {//GEN-END:|56-getter|0|56-preInit
            // write pre-init user code here
            //se accede a los json guardados en fichero
            gjson_view = new getJSON(f_rec);
            if (jsons == null)
                jsons = gjson_view.jsons();
            form_view = new Form("Ver registro diario", new Item[] { getChoiceGroup_view()
});//GEN-BEGIN:|56-getter|1|56-postInit
            form_view.addCommand(getOkCommand_view());
            form_view.addCommand(getBackCommand_view());
```

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

```
            form_view.setCommandListener(this);//GEN-END:|56-getter|1|56-postInit
            // write post-init user code here
        }//GEN-BEGIN:|56-getter|2|
        return form_view;
    }
    //</editor-fold>//GEN-END:|56-getter|2|

    //<editor-fold defaultstate="collapsed" desc=" Generated Getter: form_edit
">//GEN-BEGIN:|60-getter|0|60-preInit
    /**
     * Returns an initiliazed instance of form_edit component.
     * @return the initialized component instance
     */
    public Form getForm_edit() {
        if (form_edit == null) {//GEN-END:|60-getter|0|60-preInit
            // write pre-init user code here
            form_edit = new Form("Editar registro diario", new Item[] { getChoiceGroup_edit()
 });//GEN-BEGIN:|60-getter|1|60-postInit
            form_edit.addCommand(getOkCommand_edit());
            form_edit.addCommand(getBackCommand_edit());
            form_edit.setCommandListener(this);//GEN-END:|60-getter|1|60-postInit
            // write post-init user code here
        }//GEN-BEGIN:|60-getter|2|
        return form_edit;
    }
    //</editor-fold>//GEN-END:|60-getter|2|

    //<editor-fold defaultstate="collapsed" desc=" Generated Getter: okCommand_main
">//GEN-BEGIN:|66-getter|0|66-preInit
    /**
     * Returns an initiliazed instance of okCommand_main component.
     * @return the initialized component instance
     */
    public Command getOkCommand_main() {
        if (okCommand_main == null) {//GEN-END:|66-getter|0|66-preInit
            // write pre-init user code here
            okCommand_main = new Command("Ok", Command.OK, 0);
//GEN-LINE:|66-getter|1|66-postInit
            // write post-init user code here
        }//GEN-BEGIN:|66-getter|2|
        return okCommand_main;
    }
    //</editor-fold>//GEN-END:|66-getter|2|

    //<editor-fold defaultstate="collapsed" desc=" Generated Getter: okCommand_add
">//GEN-BEGIN:|70-getter|0|70-preInit
    /**
     * Returns an initiliazed instance of okCommand_add component.
     * @return the initialized component instance
     */
    public Command getOkCommand_add() {
        if (okCommand_add == null) {//GEN-END:|70-getter|0|70-preInit
            // write pre-init user code here
            okCommand_add = new Command("Ok", Command.OK, 0);
//GEN-LINE:|70-getter|1|70-postInit
            // write post-init user code here
        }//GEN-BEGIN:|70-getter|2|
```

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

```
        return okCommand_add;
    }
    //</editor-fold>//GEN-END:|70-getter|2|

    //<editor-fold defaultstate="collapsed" desc=" Generated Getter: backCommand_view
">//GEN-BEGIN:|74-getter|0|74-preInit
    /**
     * Returns an initiliazed instance of backCommand_view component.
     * @return the initialized component instance
     */
    public Command getBackCommand_view() {
        if (backCommand_view == null) {//GEN-END:|74-getter|0|74-preInit
            // write pre-init user code here
            backCommand_view = new Command("Back", Command.BACK, 0);
//GEN-LINE:|74-getter|1|74-postInit
            // write post-init user code here
        }//GEN-BEGIN:|74-getter|2|
        return backCommand_view;
    }
    //</editor-fold>//GEN-END:|74-getter|2|

    //<editor-fold defaultstate="collapsed" desc=" Generated Getter: backCommand_add
">//GEN-BEGIN:|76-getter|0|76-preInit
    /**
     * Returns an initiliazed instance of backCommand_add component.
     * @return the initialized component instance
     */
    public Command getBackCommand_add() {
        if (backCommand_add == null) {//GEN-END:|76-getter|0|76-preInit
            // write pre-init user code here
            backCommand_add = new Command("Back", Command.BACK, 0);
//GEN-LINE:|76-getter|1|76-postInit
            // write post-init user code here
        }//GEN-BEGIN:|76-getter|2|
        return backCommand_add;
    }
    //</editor-fold>//GEN-END:|76-getter|2|

    //<editor-fold defaultstate="collapsed" desc=" Generated Getter: okCommand_edit
">//GEN-BEGIN:|78-getter|0|78-preInit
    /**
     * Returns an initiliazed instance of okCommand_edit component.
     * @return the initialized component instance
     */
    public Command getOkCommand_edit() {
        if (okCommand_edit == null) {//GEN-END:|78-getter|0|78-preInit
            // write pre-init user code here
            okCommand_edit = new Command("Ok", Command.OK, 0);
//GEN-LINE:|78-getter|1|78-postInit
            // write post-init user code here
        }//GEN-BEGIN:|78-getter|2|
        return okCommand_edit;
    }
    //</editor-fold>//GEN-END:|78-getter|2|

    //<editor-fold defaultstate="collapsed" desc=" Generated Getter: backCommand_edit
">//GEN-BEGIN:|80-getter|0|80-preInit
```

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

```
    /**
     * Returns an initiliazed instance of backCommand_edit component.
     * @return the initialized component instance
     */
    public Command getBackCommand_edit() {
        if (backCommand_edit == null) {//GEN-END:|80-getter|0|80-preInit
            // write pre-init user code here
            backCommand_edit = new Command("Back", Command.BACK, 0);
//GEN-LINE:|80-getter|1|80-postInit
            // write post-init user code here
        }//GEN-BEGIN:|80-getter|2|
        return backCommand_edit;
    }
    //</editor-fold>//GEN-END:|80-getter|2|

    //<editor-fold defaultstate="collapsed" desc=" Generated Getter: Transfer
">//GEN-BEGIN:|107-getter|0|107-preInit
    /**
     * Returns an initiliazed instance of Transfer component.
     * @return the initialized component instance
     */
    public List getTransfer() {
        if (Transfer == null) {//GEN-END:|107-getter|0|107-preInit
            // write pre-init user code here
            Transfer = new List("Transferencia de registros", Choice.IMPLICIT);
//GEN-BEGIN:|107-getter|1|107-postInit
            Transfer.append("Transferir con HQ", null);
            Transfer.append("Transferir a tarjeta de memoria", null);
            Transfer.append("Transferir con base", null);
            Transfer.addCommand(getOkCommand_transfer());
            Transfer.addCommand(getBackCommand_transfer());
            Transfer.setCommandListener(this);
            Transfer.setSelectedFlags(new boolean[] { false, false, false });
//GEN-END:|107-getter|1|107-postInit
            // write post-init user code here
        }//GEN-BEGIN:|107-getter|2|
        return Transfer;
    }
    //</editor-fold>//GEN-END:|107-getter|2|

    //<editor-fold defaultstate="collapsed" desc=" Generated Method: TransferAction
">//GEN-BEGIN:|107-action|0|107-preAction
    /**
     * Performs an action assigned to the selected list element in the Transfer component.
     */
    public void TransferAction() {//GEN-END:|107-action|0|107-preAction
        // enter pre-action user code here
        String __selectedString = getTransfer().getString(getTransfer().getSelectedIndex());
//GEN-BEGIN:|107-action|1|115-preAction
        if (__selectedString != null) {
            if (__selectedString.equals("Transferir con HQ")) {
//GEN-END:|107-action|1|115-preAction
                // write pre-action user code here
                switchDisplayable(null, getWaitScreen_HQ());
//GEN-LINE:|107-action|2|115-postAction
                // write post-action user code here
            } else if (__selectedString.equals("Transferir a tarjeta de memoria")) {
```

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

```
//GEN-LINE:|107-action|3|117-preAction
                // write pre-action user code here
                /*
switchDisplayable (null, getWaitScreen_SD ());//GEN-LINE:|107-action|4|117-postAction
                */
                // write post-action user code here
            } else if (__selectedString.equals("Transferir con base")) {
//GEN-LINE:|107-action|5|114-preAction
                // write pre-action user code here
                /*
switchDisplayable (null, getWaitScreen_Base ());//GEN-LINE:|107-action|6|114-postAction
                */
                // write post-action user code here
            }//GEN-BEGIN:|107-action|7|107-postAction
        }//GEN-END:|107-action|7|107-postAction
        // enter post-action user code here
    }//GEN-BEGIN:|107-action|8|
    //</editor-fold>//GEN-END:|107-action|8|

    //<editor-fold defaultstate="collapsed" desc=" Generated Getter: backCommand_transfer
">//GEN-BEGIN:|112-getter|0|112-preInit
    /**
     * Returns an initiliazed instance of backCommand_transfer component.
     * @return the initialized component instance
     */
    public Command getBackCommand_transfer() {
        if (backCommand_transfer == null) {//GEN-END:|112-getter|0|112-preInit
            // write pre-init user code here
            backCommand_transfer = new Command("Back", Command.BACK, 0);
//GEN-LINE:|112-getter|1|112-postInit
            // write post-init user code here
        }//GEN-BEGIN:|112-getter|2|
        return backCommand_transfer;
    }
    //</editor-fold>//GEN-END:|112-getter|2|

    //<editor-fold defaultstate="collapsed" desc=" Generated Getter: settings
">//GEN-BEGIN:|140-getter|0|140-preInit
    /**
     * Returns an initiliazed instance of settings component.
     * @return the initialized component instance
     */
    public List getSettings() {
        if (settings == null) {//GEN-END:|140-getter|0|140-preInit
            // write pre-init user code here
            settings = new List("Settings", Choice.IMPLICIT);
//GEN-BEGIN:|140-getter|1|140-postInit
            settings.append("Ajustes de dispositivo", null);
            settings.append("Aplicar ajustes a datos almacenados", null);
            settings.addCommand(getOkCommand_config());
            settings.addCommand(getBackCommand_config());
            settings.setCommandListener(this);
            settings.setSelectedFlags(new boolean[] { false, false });
//GEN-END:|140-getter|1|140-postInit
            // write post-init user code here
        }//GEN-BEGIN:|140-getter|2|
        return settings;
```

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

```
    }
    //</editor-fold>//GEN-END:|140-getter|2|

    //<editor-fold defaultstate="collapsed" desc=" Generated Method: settingsAction
">//GEN-BEGIN:|140-action|0|140-preAction
    /**
     * Performs an action assigned to the selected list element in the settings component.
     */
    public void settingsAction() {//GEN-END:|140-action|0|140-preAction
        // enter pre-action user code here
        String __selectedString = getSettings().getString(getSettings().getSelectedIndex());
//GEN-BEGIN:|140-action|1|144-preAction
        if (__selectedString != null) {
            if (__selectedString.equals("Ajustes de dispositivo")) {
//GEN-END:|140-action|1|144-preAction
                // write pre-action user code here
                switchDisplayable(null, getForm_settings());
//GEN-LINE:|140-action|2|144-postAction
                // write post-action user code here
            } else if (__selectedString.equals("Aplicar ajustes a datos almacenados")) {
//GEN-LINE:|140-action|3|143-preAction
                // write pre-action user code here
                switchDisplayable(null, getMain());//GEN-LINE:|140-action|4|143-postAction
                // write post-action user code here
            }//GEN-BEGIN:|140-action|5|140-postAction
        }//GEN-END:|140-action|5|140-postAction
        // enter post-action user code here
    }//GEN-BEGIN:|140-action|6|
    //</editor-fold>//GEN-END:|140-action|6|

    //<editor-fold defaultstate="collapsed" desc=" Generated Getter: backCommand_config
">//GEN-BEGIN:|170-getter|0|170-preInit
    /**
     * Returns an initiliazed instance of backCommand_config component.
     * @return the initialized component instance
     */
    public Command getBackCommand_config() {
        if (backCommand_config == null) {//GEN-END:|170-getter|0|170-preInit
            // write pre-init user code here
            backCommand_config = new Command("Back", Command.BACK, 0);
//GEN-LINE:|170-getter|1|170-postInit
            // write post-init user code here
        }//GEN-BEGIN:|170-getter|2|
        return backCommand_config;
    }
    //</editor-fold>//GEN-END:|170-getter|2|

    //<editor-fold defaultstate="collapsed" desc=" Generated Getter: choiceGroup_symptom
">//GEN-BEGIN:|225-getter|0|225-preInit
    /**
     * Returns an initiliazed instance of choiceGroup_symptom component.
     * @return the initialized component instance
     */
    public ChoiceGroup getChoiceGroup_symptom() {
        if (choiceGroup_symptom == null) {//GEN-END:|225-getter|0|225-preInit
            // write pre-init user code here
            //aquí generamos la lista de síntomas en función al fichero de configuración
```

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

```
            choiceGroup_symptom = new ChoiceGroup("choiceGroup", Choice.EXCLUSIVE);
            for (int i=0;i<number_of_diseases;i++) {
                choiceGroup_symptom.append(diseases[i], null);
            }
            for (int i=0;i<number_of_diseases;i++) {
                choiceGroup_symptom.setFont(i, null);
            }
            choiceGroup_symptom.setFitPolicy(Choice.TEXT_WRAP_DEFAULT);
            booleano = new boolean[number_of_diseases];
            for (int i=0;i<number_of_diseases;i++) {
                booleano[i] = false;
            }
            choiceGroup_symptom.setSelectedFlags(booleano);
/* código generado por NetBeans
choiceGroup_symptom = new ChoiceGroup ("Seleccione s\u00EDntoma",
Choice.EXCLUSIVE);//GEN-BEGIN:|225-getter|1|225-postInit
choiceGroup_symptom.append ("Choice Element 1", null);
choiceGroup_symptom.setLayout (ImageItem.LAYOUT_DEFAULT);
choiceGroup_symptom.setFitPolicy (Choice.TEXT_WRAP_DEFAULT);
choiceGroup_symptom.setSelectedFlags (new boolean[] { false
});//GEN-END:|225-getter|1|225-postInit
*/            // write post-init user code here*/
        }//GEN-BEGIN:|225-getter|2|
        return choiceGroup_symptom;
    }
    //</editor-fold>//GEN-END:|225-getter|2|

    //<editor-fold defaultstate="collapsed" desc=" Generated Getter:
backCommand_select_symptom ">//GEN-BEGIN:|229-getter|0|229-preInit
    /**
     * Returns an initiliazed instance of backCommand_select_symptom component.
     * @return the initialized component instance
     */
    public Command getBackCommand_select_symptom() {
        if (backCommand_select_symptom == null) {//GEN-END:|229-getter|0|229-preInit
            // write pre-init user code here
            backCommand_select_symptom = new Command("Back", Command.BACK, 0);
//GEN-LINE:|229-getter|1|229-postInit
            // write post-init user code here
        }//GEN-BEGIN:|229-getter|2|
        return backCommand_select_symptom;
    }
    //</editor-fold>//GEN-END:|229-getter|2|

    //<editor-fold defaultstate="collapsed" desc=" Generated Getter: form_add_number
">//GEN-BEGIN:|253-getter|0|253-preInit
    /**
     * Returns an initiliazed instance of form_add_number component.
     * @return the initialized component instance
     */
    public Form getForm_add_number() {
        if (form_add_number == null) {//GEN-END:|253-getter|0|253-preInit
            // write pre-init user code here
            form_add_number = new Form("Introduzca el n\u00FAmero de casos:", new Item[] {
getTextField_mu5(), getTextField_mo5(), getTextField_fu5(), getTextField_fo5() });
//GEN-BEGIN:|253-getter|1|253-postInit
            form_add_number.addCommand(getOkCommand_number());
```

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

```
            form_add_number.addCommand(getBackCommand_number());
            form_add_number.setCommandListener(this);//GEN-END:|253-getter|1|253-postInit
            // write post-init user code here
        }//GEN-BEGIN:|253-getter|2|
        return form_add_number;
    }
    //</editor-fold>//GEN-END:|253-getter|2|

    //<editor-fold defaultstate="collapsed" desc=" Generated Getter: textField_mu5
">//GEN-BEGIN:|254-getter|0|254-preInit
    /**
     * Returns an initiliazed instance of textField_mu5 component.
     * @return the initialized component instance
     */
    public TextField getTextField_mu5() {
        if (textField_mu5 == null) {//GEN-END:|254-getter|0|254-preInit
            // write pre-init user code here
            textField_mu5 = new TextField("Male under 5 years", null, 32, TextField.NUMERIC);
//GEN-LINE:|254-getter|1|254-postInit
            // write post-init user code here
        }//GEN-BEGIN:|254-getter|2|
        return textField_mu5;
    }
    //</editor-fold>//GEN-END:|254-getter|2|

    //<editor-fold defaultstate="collapsed" desc=" Generated Getter: okCommand_number
">//GEN-BEGIN:|255-getter|0|255-preInit
    /**
     * Returns an initiliazed instance of okCommand_number component.
     * @return the initialized component instance
     */
    public Command getOkCommand_number() {
        if (okCommand_number == null) {//GEN-END:|255-getter|0|255-preInit
            // write pre-init user code here
            okCommand_number = new Command("Ok", Command.OK, 0);
//GEN-LINE:|255-getter|1|255-postInit
            // write post-init user code here
        }//GEN-BEGIN:|255-getter|2|
        return okCommand_number;
    }
    //</editor-fold>//GEN-END:|255-getter|2|

    //<editor-fold defaultstate="collapsed" desc=" Generated Getter:
okCommand_select_symptom ">//GEN-BEGIN:|291-getter|0|291-preInit
    /**
     * Returns an initiliazed instance of okCommand_select_symptom component.
     * @return the initialized component instance
     */
    public Command getOkCommand_select_symptom() {
        if (okCommand_select_symptom == null) {//GEN-END:|291-getter|0|291-preInit
            // write pre-init user code here
            okCommand_select_symptom = new Command("Ok", Command.OK, 0);
//GEN-LINE:|291-getter|1|291-postInit
            // write post-init user code here
        }//GEN-BEGIN:|291-getter|2|
        return okCommand_select_symptom;
```

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

```
    //</editor-fold>//GEN-END:|291-getter|2|

    //<editor-fold defaultstate="collapsed" desc=" Generated Getter: textField_mo5
">//GEN-BEGIN:|314-getter|0|314-preInit
    /**
     * Returns an initiliazed instance of textField_mo5 component.
     * @return the initialized component instance
     */
    public TextField getTextField_mo5() {
        if (textField_mo5 == null) {//GEN-END:|314-getter|0|314-preInit
            // write pre-init user code here
            textField_mo5 = new TextField("Male over 5 years", null, 32, TextField.NUMERIC);
//GEN-LINE:|314-getter|1|314-postInit
            // write post-init user code here
        }//GEN-BEGIN:|314-getter|2|
        return textField_mo5;
    }
    //</editor-fold>//GEN-END:|314-getter|2|

    //<editor-fold defaultstate="collapsed" desc=" Generated Getter: textField_fu5
">//GEN-BEGIN:|315-getter|0|315-preInit
    /**
     * Returns an initiliazed instance of textField_fu5 component.
     * @return the initialized component instance
     */
    public TextField getTextField_fu5() {
        if (textField_fu5 == null) {//GEN-END:|315-getter|0|315-preInit
            // write pre-init user code here
            textField_fu5 = new TextField("Female under 5 years", null, 32, TextField.NUMERIC
);//GEN-LINE:|315-getter|1|315-postInit
            // write post-init user code here
        }//GEN-BEGIN:|315-getter|2|
        return textField_fu5;
    }
    //</editor-fold>//GEN-END:|315-getter|2|

    //<editor-fold defaultstate="collapsed" desc=" Generated Getter: textField_fo5
">//GEN-BEGIN:|316-getter|0|316-preInit
    /**
     * Returns an initiliazed instance of textField_fo5 component.
     * @return the initialized component instance
     */
    public TextField getTextField_fo5() {
        if (textField_fo5 == null) {//GEN-END:|316-getter|0|316-preInit
            // write pre-init user code here
            textField_fo5 = new TextField("Female over 5 years", null, 32, TextField.NUMERIC
);//GEN-LINE:|316-getter|1|316-postInit
            // write post-init user code here
        }//GEN-BEGIN:|316-getter|2|
        return textField_fo5;
    }
    //</editor-fold>//GEN-END:|316-getter|2|

    //<editor-fold defaultstate="collapsed" desc=" Generated Getter: backCommand_number
">//GEN-BEGIN:|333-getter|0|333-preInit
    /**
     * Returns an initiliazed instance of backCommand_number component.
```

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

```
     * @return the initialized component instance
     */
    public Command getBackCommand_number() {
        if (backCommand_number == null) {//GEN-END:|333-getter|0|333-preInit
            // write pre-init user code here
            backCommand_number = new Command("Back", Command.BACK, 0);
//GEN-LINE:|333-getter|1|333-postInit
            // write post-init user code here
        }//GEN-BEGIN:|333-getter|2|
        return backCommand_number;
    }
    //</editor-fold>//GEN-END:|333-getter|2|

    //<editor-fold defaultstate="collapsed" desc=" Generated Getter: textBox_add
">//GEN-BEGIN:|351-getter|0|351-preInit
    /**
     * Returns an initiliazed instance of textBox_add component.
     * @return the initialized component instance
     */
    public TextBox getTextBox_add() {
        if (textBox_add == null) {//GEN-END:|351-getter|0|351-preInit
            // write pre-init user code here
            //guarda en las variables los datos recogidos en el formulario
            if (data == null) {
                data = new String[4][number_of_diseases];
                for (int i = 0; i < 4; i++)
                    for (int j = 0; j < number_of_diseases; j++)
                        data[i][j] = "0";
            }
            date_add = generaDate(dateField_add.getDate());//guarda en date_add la fecha
introducida
            disease = choiceGroup_symptom.getSelectedIndex();//guarda el síntoma seleccionado
            if (textField_mu5.getString().compareTo("") != 0)
                data[0][disease] = textField_mu5.getString();
            if (textField_mo5.getString().compareTo("") != 0)
                data[1][disease] = textField_mo5.getString();
            if (textField_fu5.getString().compareTo("") != 0)
                data[2][disease] = textField_fu5.getString();
            if (textField_fo5.getString().compareTo("") != 0)
                data[3][disease] = textField_fo5.getString();
            //genera "veamos": string que se representa en el textbox siguiente
            veamos = "Date: "+date_add+"\n";
            for (int i=0;i<4;i++) {
                if (i == 0)
                    veamos += "Male under 5:\n";
                else if (i == 1)
                    veamos += "Male over 5:\n";
                else if (i == 2)
                    veamos += "Female under 5:\n";
                else if (i == 3)
                    veamos += "Female over 5:\n";
                for (int j=0;j<number_of_diseases;j++) {
                    veamos += data[i][j]+", ";
                }
                veamos += "\n";
            }
            textBox_add = new TextBox("Registro generado", veamos, 1000, TextField.ANY |
```

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

```
TextField.UNEDITABLE);//GEN-BEGIN:|351-getter|1|351-postInit
            textBox_add.addCommand(getOkCommand_textBox_add());
            textBox_add.setCommandListener(this);//GEN-END:|351-getter|1|351-postInit
            // write post-init user code here
        }//GEN-BEGIN:|351-getter|2|
        return textBox_add;
    }
    //</editor-fold>//GEN-END:|351-getter|2|

    //<editor-fold defaultstate="collapsed" desc=" Generated Getter: okCommand_textBox_add
">//GEN-BEGIN:|359-getter|0|359-preInit
    /**
     * Returns an initiliazed instance of okCommand_textBox_add component.
     * @return the initialized component instance
     */
    public Command getOkCommand_textBox_add() {
        if (okCommand_textBox_add == null) {//GEN-END:|359-getter|0|359-preInit
            // write pre-init user code here
            okCommand_textBox_add = new Command("Ok", Command.OK, 0);
//GEN-LINE:|359-getter|1|359-postInit
            // write post-init user code here
        }//GEN-BEGIN:|359-getter|2|
        return okCommand_textBox_add;
    }
    //</editor-fold>//GEN-END:|359-getter|2|

    //<editor-fold defaultstate="collapsed" desc=" Generated Getter: form_add_select_symptom
">//GEN-BEGIN:|221-getter|0|221-preInit
    /**
     * Returns an initiliazed instance of form_add_select_symptom component.
     * @return the initialized component instance
     */
    public Form getForm_add_select_symptom() {
        if (form_add_select_symptom == null) {//GEN-END:|221-getter|0|221-preInit
            // write pre-init user code here
            form_add_select_symptom = new Form("A\u00F1adir registro", new Item[] {
getChoiceGroup_symptom() });//GEN-BEGIN:|221-getter|1|221-postInit
            form_add_select_symptom.addCommand(getOkCommand_select_symptom());
            form_add_select_symptom.addCommand(getBackCommand_select_symptom());
            form_add_select_symptom.setCommandListener(this);
//GEN-END:|221-getter|1|221-postInit
            // write post-init user code here
        }//GEN-BEGIN:|221-getter|2|
        return form_add_select_symptom;
    }
    //</editor-fold>//GEN-END:|221-getter|2|

    //<editor-fold defaultstate="collapsed" desc=" Generated Getter: okCommand_bucle_add
">//GEN-BEGIN:|364-getter|0|364-preInit
    /**
     * Returns an initiliazed instance of okCommand_bucle_add component.
     * @return the initialized component instance
     */
    public Command getOkCommand_bucle_add() {
        if (okCommand_bucle_add == null) {//GEN-END:|364-getter|0|364-preInit
            // write pre-init user code here
            okCommand_bucle_add = new Command("Ok", Command.OK, 0);
```

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

```
//GEN-LINE:|364-getter|1|364-postInit
            // write post-init user code here
        }//GEN-BEGIN:|364-getter|2|
        return okCommand_bucle_add;
    }
    //</editor-fold>//GEN-END:|364-getter|2|

    //<editor-fold defaultstate="collapsed" desc=" Generated Getter: bucle_add
">//GEN-BEGIN:|368-getter|0|368-preInit
    /**
     * Returns an initiliazed instance of bucle_add component.
     * @return the initialized component instance
     */
    public List getBucle_add() {
        if (bucle_add == null) {//GEN-END:|368-getter|0|368-preInit
            // write pre-init user code here
            bucle_add = new List("Elija una opci\u00F3n:", Choice.IMPLICIT);
//GEN-BEGIN:|368-getter|1|368-postInit
            bucle_add.append("A\u00F1adir m\u00E1s s\u00EDntomas del mismo d\u00EDa", null);
            bucle_add.append("Guardar y volver al Men\u00FA principal", null);
            bucle_add.addCommand(getOkCommand_bucle_add());
            bucle_add.setCommandListener(this);
            bucle_add.setSelectedFlags(new boolean[] { false, false });
//GEN-END:|368-getter|1|368-postInit
            // write post-init user code here
        }//GEN-BEGIN:|368-getter|2|
        return bucle_add;
    }
    //</editor-fold>//GEN-END:|368-getter|2|

    //<editor-fold defaultstate="collapsed" desc=" Generated Method: bucle_addAction
">//GEN-BEGIN:|368-action|0|368-preAction
    /**
     * Performs an action assigned to the selected list element in the bucle_add component.
     */
    public void bucle_addAction() {//GEN-END:|368-action|0|368-preAction
        // enter pre-action user code here
        String __selectedString = getBucle_add().getString(getBucle_add().getSelectedIndex
());//GEN-BEGIN:|368-action|1|373-preAction
        if (__selectedString != null) {
            if (__selectedString.equals("A\u00F1adir m\u00E1s s\u00EDntomas del mismo
d\u00EDa")) {//GEN-END:|368-action|1|373-preAction
                // write pre-action user code here
                resetForm_add();
                if (choiceGroup_symptom != null)
                    choiceGroup_symptom.setSelectedIndex(disease+1, true);
                switchDisplayable(null, getForm_add_select_symptom());
//GEN-LINE:|368-action|2|373-postAction
                // write post-action user code here
            } else if (__selectedString.equals("Guardar y volver al Men\u00FA principal")) {
//GEN-LINE:|368-action|3|374-preAction
                // write pre-action user code here
                stringjson = generaJSON();
                saveData sd = new saveData();
                exito = sd.saveData(f_rec, stringjson, false);
                data = null;
                vjson_del = vjson_edit = vjson_view = null;
```

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

```
                choiceGroup_del = choiceGroup_edit = choiceGroup_view = null;
                choiceGroup_symptom.setSelectedIndex(0, true);
                switchDisplayable(null, getMain());//GEN-LINE:|368-action|4|374-postAction
                // write post-action user code here
            }//GEN-BEGIN:|368-action|5|368-postAction
        }//GEN-END:|368-action|5|368-postAction
        // enter post-action user code here
    }//GEN-BEGIN:|368-action|6|
    //</editor-fold>//GEN-END:|368-action|6|

    //<editor-fold defaultstate="collapsed" desc=" Generated Getter: textBox_view
">//GEN-BEGIN:|379-getter|0|379-preInit
    /**
     * Returns an initiliazed instance of textBox_view component.
     * @return the initialized component instance
     */
    public TextBox getTextBox_view() {
        if (textBox_view == null) {//GEN-END:|379-getter|0|379-preInit
            // write pre-init user code here
            //muestra los datos del registro seleccionado para visualizar
            JSONArray arr_view_mu5 = new JSONArray();
            JSONArray arr_view_mo5 = new JSONArray();
            JSONArray arr_view_fu5 = new JSONArray();
            JSONArray arr_view_fo5 = new JSONArray();
            try {
                json_view = jObject[choiceGroup_view.getSelectedIndex()];
                arr_view_mu5 = json_view.getJSONArray("mu5");
                arr_view_mo5 = json_view.getJSONArray("mo5");
                arr_view_fu5 = json_view.getJSONArray("fu5");
                arr_view_fo5 = json_view.getJSONArray("fo5");
                date_view = json_view.getString("date");

                //genera "str_view": string que se representa en el textbox siguiente
                str_view = "Date: "+date_view+"\n";
                for (int i=0; i<4; i++) {
                    if (i == 0) {
                        str_view += "male under 5:\n";
                        for (int j=0; j<arr_view_mu5.length(); j++)
                            str_view += arr_view_mu5.getInt(j)+",";
                        str_view = str_view.substring(0, str_view.length()-1);//quitar
última coma ,
                    }
                    else if (i == 1) {
                        str_view += "male over 5:\n";
                        for (int k=0; k<arr_view_mo5.length(); k++)
                            str_view += arr_view_mo5.getInt(k)+",";
                        str_view = str_view.substring(0, str_view.length()-1);//quitar
última coma ,
                    }
                    else if (i == 2) {
                        str_view += "female under 5:\n";
                        for (int l=0; l<arr_view_fu5.length(); l++)
                            str_view += arr_view_fu5.getInt(l)+",";
                        str_view = str_view.substring(0, str_view.length()-1);//quitar
última coma ,
                    }
                    else if (i == 3) {
```

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

```
                        str_view += "female over 5:\n";
                        for (int m=0; m<arr_view_fo5.length(); m++)
                            str_view += arr_view_fo5.getInt(m)+",";
                        str_view = str_view.substring(0, str_view.length()-1);//quitar
última coma ,
                    }
                    str_view += "\n";
                }
            }
            catch (JSONException jse) {
                jse.printStackTrace();
            }
            catch (java.lang.ArrayIndexOutOfBoundsException e) {
                switchDisplayable(null, getMain());
            }

            textBox_view = new TextBox("Salida:", str_view, 10000, TextField.ANY | TextField.
UNEDITABLE);//GEN-BEGIN:|379-getter|1|379-postInit
            textBox_view.addCommand(getOkCommand_textBox_view());
            textBox_view.addCommand(getBackCommand_textBox_view());
            textBox_view.setCommandListener(this);//GEN-END:|379-getter|1|379-postInit
            // write post-init user code here
        }//GEN-BEGIN:|379-getter|2|
        return textBox_view;
    }
    //</editor-fold>//GEN-END:|379-getter|2|

    //<editor-fold defaultstate="collapsed" desc=" Generated Getter: okCommand_view
">//GEN-BEGIN:|377-getter|0|377-preInit
    /**
     * Returns an initiliazed instance of okCommand_view component.
     * @return the initialized component instance
     */
    public Command getOkCommand_view() {
        if (okCommand_view == null) {//GEN-END:|377-getter|0|377-preInit
            // write pre-init user code here
            okCommand_view = new Command("Ok", Command.OK, 0);
//GEN-LINE:|377-getter|1|377-postInit
            // write post-init user code here
        }//GEN-BEGIN:|377-getter|2|
        return okCommand_view;
    }
    //</editor-fold>//GEN-END:|377-getter|2|

    //<editor-fold defaultstate="collapsed" desc=" Generated Getter: okCommand_textBox_view
">//GEN-BEGIN:|381-getter|0|381-preInit
    /**
     * Returns an initiliazed instance of okCommand_textBox_view component.
     * @return the initialized component instance
     */
    public Command getOkCommand_textBox_view() {
        if (okCommand_textBox_view == null) {//GEN-END:|381-getter|0|381-preInit
            // write pre-init user code here
            okCommand_textBox_view = new Command("Ok", Command.OK, 0);
//GEN-LINE:|381-getter|1|381-postInit
            // write post-init user code here
        }//GEN-BEGIN:|381-getter|2|
```

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

```
        return okCommand_textBox_view;
    }
    //</editor-fold>//GEN-END:|381-getter|2|

    //<editor-fold defaultstate="collapsed" desc=" Generated Getter:
backCommand_textBox_view ">//GEN-BEGIN:|384-getter|0|384-preInit
    /**
     * Returns an initiliazed instance of backCommand_textBox_view component.
     * @return the initialized component instance
     */
    public Command getBackCommand_textBox_view() {
        if (backCommand_textBox_view == null) {//GEN-END:|384-getter|0|384-preInit
            // write pre-init user code here
            backCommand_textBox_view = new Command("Back", Command.BACK, 0);
//GEN-LINE:|384-getter|1|384-postInit
            // write post-init user code here
        }//GEN-BEGIN:|384-getter|2|
        return backCommand_textBox_view;
    }
    //</editor-fold>//GEN-END:|384-getter|2|

    //<editor-fold defaultstate="collapsed" desc=" Generated Getter: okCommand_config
">//GEN-BEGIN:|387-getter|0|387-preInit
    /**
     * Returns an initiliazed instance of okCommand_config component.
     * @return the initialized component instance
     */
    public Command getOkCommand_config() {
        if (okCommand_config == null) {//GEN-END:|387-getter|0|387-preInit
            // write pre-init user code here
            okCommand_config = new Command("Ok", Command.OK, 0);
//GEN-LINE:|387-getter|1|387-postInit
            // write post-init user code here
        }//GEN-BEGIN:|387-getter|2|
        return okCommand_config;
    }
    //</editor-fold>//GEN-END:|387-getter|2|

    //<editor-fold defaultstate="collapsed" desc=" Generated Getter: form_del
">//GEN-BEGIN:|408-getter|0|408-preInit
    /**
     * Returns an initiliazed instance of form_del component.
     * @return the initialized component instance
     */
    public Form getForm_del() {
        if (form_del == null) {//GEN-END:|408-getter|0|408-preInit
            // write pre-init user code here
            // Eliminar registro. Primero se cargan los registros del fichero
            // pero se visualizan sólo las fechas. La fecha elegida es el registro que se
borrará
            // Se sobreescribirá el fichero con los strings restantes
            form_del = new Form("Eliminar registro diario", new Item[] { getChoiceGroup_del()
 });//GEN-BEGIN:|408-getter|1|408-postInit
            form_del.addCommand(getOkCommand_del());
            form_del.addCommand(getBackCommand_del());
            form_del.setCommandListener(this);//GEN-END:|408-getter|1|408-postInit
            // write post-init user code here
```

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

```
        }//GEN-BEGIN:|408-getter|2|
        return form_del;
    }
    //</editor-fold>//GEN-END:|408-getter|2|

    //<editor-fold defaultstate="collapsed" desc=" Generated Getter: backCommand_del
">//GEN-BEGIN:|409-getter|0|409-preInit
    /**
     * Returns an initiliazed instance of backCommand_del component.
     * @return the initialized component instance
     */
    public Command getBackCommand_del() {
        if (backCommand_del == null) {//GEN-END:|409-getter|0|409-preInit
            // write pre-init user code here
            backCommand_del = new Command("Back", Command.BACK, 0);
//GEN-LINE:|409-getter|1|409-postInit
            // write post-init user code here
        }//GEN-BEGIN:|409-getter|2|
        return backCommand_del;
    }
    //</editor-fold>//GEN-END:|409-getter|2|

    //<editor-fold defaultstate="collapsed" desc=" Generated Getter: okCommand_del
">//GEN-BEGIN:|411-getter|0|411-preInit
    /**
     * Returns an initiliazed instance of okCommand_del component.
     * @return the initialized component instance
     */
    public Command getOkCommand_del() {
        if (okCommand_del == null) {//GEN-END:|411-getter|0|411-preInit
            // write pre-init user code here
            okCommand_del = new Command("Ok", Command.OK, 0);
//GEN-LINE:|411-getter|1|411-postInit
            // write post-init user code here
        }//GEN-BEGIN:|411-getter|2|
        return okCommand_del;
    }
    //</editor-fold>//GEN-END:|411-getter|2|

    //<editor-fold defaultstate="collapsed" desc=" Generated Getter: choiceGroup_del
">//GEN-BEGIN:|414-getter|0|414-preInit
    /**
     * Returns an initiliazed instance of choiceGroup_del component.
     * @return the initialized component instance
     */
    public ChoiceGroup getChoiceGroup_del() {
        if (choiceGroup_del == null) {//GEN-END:|414-getter|0|414-preInit
            // write pre-init user code here
            /*
choiceGroup_del = new ChoiceGroup ("Registros disponibles:",
Choice.EXCLUSIVE);//GEN-LINE:|414-getter|1|414-postInit
            // write post-init user code here*/
            choiceGroup_del = generaChoiceGroup(vjson_del, choiceGroup_del);
        }//GEN-BEGIN:|414-getter|2|
        return choiceGroup_del;
    }
    //</editor-fold>//GEN-END:|414-getter|2|
```

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

```
    //<editor-fold defaultstate="collapsed" desc=" Generated Getter: form_settings
">//GEN-BEGIN:|428-getter|0|428-preInit
    /**
     * Returns an initiliazed instance of form_settings component.
     * @return the initialized component instance
     */
    public Form getForm_settings() {
        if (form_settings == null) {//GEN-END:|428-getter|0|428-preInit
            // write pre-init user code here
            form_settings = new Form("Ajustes actuales:", new Item[] { getID_Op_field(),
getID_Base_field(), getID_Device_field() });//GEN-BEGIN:|428-getter|1|428-postInit
            form_settings.addCommand(getOkCommand_form_set());
            form_settings.setCommandListener(this);//GEN-END:|428-getter|1|428-postInit
            // write post-init user code here
        }//GEN-BEGIN:|428-getter|2|
        return form_settings;
    }
    //</editor-fold>//GEN-END:|428-getter|2|

    //<editor-fold defaultstate="collapsed" desc=" Generated Getter: ID_Op_field
">//GEN-BEGIN:|437-getter|0|437-preInit
    /**
     * Returns an initiliazed instance of ID_Op_field component.
     * @return the initialized component instance
     */
    public TextField getID_Op_field() {
        if (ID_Op_field == null) {//GEN-END:|437-getter|0|437-preInit
            // write pre-init user code here
            //try {
            ID_Op_field = new TextField("ID Operaci\u00F3n", ids[0]
//GEN-BEGIN:|437-getter|1|437-postInit
                    , 32, TextField.ANY);//GEN-END:|437-getter|1|437-postInit
            // write post-init user code here
            //} catch ( JSONException jsone) { jsone.printStackTrace(); }
        }//GEN-BEGIN:|437-getter|2|
        return ID_Op_field;
    }
    //</editor-fold>//GEN-END:|437-getter|2|

    //<editor-fold defaultstate="collapsed" desc=" Generated Getter: ID_Base_field
">//GEN-BEGIN:|438-getter|0|438-preInit
    /**
     * Returns an initiliazed instance of ID_Base_field component.
     * @return the initialized component instance
     */
    public TextField getID_Base_field() {
        if (ID_Base_field == null) {//GEN-END:|438-getter|0|438-preInit
            // write pre-init user code here
            //try {
            ID_Base_field = new TextField("ID Base", ids[1], 32, TextField.ANY);
//GEN-LINE:|438-getter|1|438-postInit
            // write post-init user code here
            //} catch ( JSONException jsone) { jsone.printStackTrace(); }
        }//GEN-BEGIN:|438-getter|2|
        return ID_Base_field;
    }
```

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

```
    //</editor-fold>//GEN-END:|438-getter|2|

    //<editor-fold defaultstate="collapsed" desc=" Generated Getter: ID_Device_field
">//GEN-BEGIN:|439-getter|0|439-preInit
    /**
     * Returns an initiliazed instance of ID_Device_field component.
     * @return the initialized component instance
     */
    public TextField getID_Device_field() {
        if (ID_Device_field == null) {//GEN-END:|439-getter|0|439-preInit
            // write pre-init user code here
            //try {
            ID_Device_field = new TextField("ID Dispositivo", ids[2], 32, TextField.ANY);
//GEN-LINE:|439-getter|1|439-postInit
            // write post-init user code here
            //} catch ( JSONException jsone) { jsone.printStackTrace(); }
        }//GEN-BEGIN:|439-getter|2|
        return ID_Device_field;
    }
    //</editor-fold>//GEN-END:|439-getter|2|

    //<editor-fold defaultstate="collapsed" desc=" Generated Getter: okCommand_form_set
">//GEN-BEGIN:|431-getter|0|431-preInit
    /**
     * Returns an initiliazed instance of okCommand_form_set component.
     * @return the initialized component instance
     */
    public Command getOkCommand_form_set() {
        if (okCommand_form_set == null) {//GEN-END:|431-getter|0|431-preInit
            // write pre-init user code here
            okCommand_form_set = new Command("Guardar", Command.OK, 0);
//GEN-LINE:|431-getter|1|431-postInit
            // write post-init user code here
        }//GEN-BEGIN:|431-getter|2|
        return okCommand_form_set;
    }
    //</editor-fold>//GEN-END:|431-getter|2|

    //<editor-fold defaultstate="collapsed" desc=" Generated Getter: textBox_del
">//GEN-BEGIN:|440-getter|0|440-preInit
    /**
     * Returns an initiliazed instance of textBox_del component.
     * @return the initialized component instance
     */
    public TextBox getTextBox_del() {
        if (textBox_del == null) {//GEN-END:|440-getter|0|440-preInit
            // write pre-init user code here
            //muestra los datos del registro seleccionado para borrar
            JSONArray arr_del_mu5 = new JSONArray();
            JSONArray arr_del_mo5 = new JSONArray();
            JSONArray arr_del_fu5 = new JSONArray();
            JSONArray arr_del_fo5 = new JSONArray();
            try {
                json_del = jObject[choiceGroup_del.getSelectedIndex()];
                arr_del_mu5 = json_del.getJSONArray("mu5");
                arr_del_mo5 = json_del.getJSONArray("mo5");
                arr_del_fu5 = json_del.getJSONArray("fu5");
```

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

```
                arr_del_fo5 = json_del.getJSONArray("fo5");
                date_del = json_del.getString("date");

                //genera "str_del": string que se representa en el textbox siguiente
                str_del = "Date: "+date_del+"\n";
                for (int i=0; i<4; i++) {
                    if (i == 0) {
                        str_del += "male under 5:\n";
                        for (int j=0; j<arr_del_mu5.length(); j++)
                            str_del += arr_del_mu5.getInt(j)+",";
                        str_del = str_del.substring(0, str_del.length()-1);//quitar última
coma ,
                    }
                    else if (i == 1) {
                        str_del += "male over 5:\n";
                        for (int k=0; k<arr_del_mo5.length(); k++)
                            str_del += arr_del_mo5.getInt(k)+",";
                        str_del = str_del.substring(0, str_del.length()-1);//quitar última
coma ,
                    }
                    else if (i == 2) {
                        str_del += "female under 5:\n";
                        for (int l=0; l<arr_del_fu5.length(); l++)
                            str_del += arr_del_fu5.getInt(l)+",";
                        str_del = str_del.substring(0, str_del.length()-1);//quitar última
coma ,
                    }
                    else if (i == 3) {
                        str_del += "female over 5:\n";
                        for (int m=0; m<arr_del_fo5.length(); m++)
                            str_del += arr_del_fo5.getInt(m)+",";
                        str_del = str_del.substring(0, str_del.length()-1);//quitar última
coma ,
                    }
                    str_del += "\n";
                }
            }
            catch (JSONException jse) {
                jse.printStackTrace();
            }
            catch (java.lang.ArrayIndexOutOfBoundsException e) {
                e.printStackTrace();
                switchDisplayable(null, getMain());
            }

            textBox_del = new TextBox("textBox_del", str_del, 1000, TextField.ANY | TextField
.UNEDITABLE);//GEN-BEGIN:|440-getter|1|440-postInit
            textBox_del.addCommand(getOkCommand_textBox_del());
            textBox_del.addCommand(getBackCommand_textBox_del());
            textBox_del.setCommandListener(this);//GEN-END:|440-getter|1|440-postInit
            // write post-init user code here
        }//GEN-BEGIN:|440-getter|2|
        return textBox_del;
    }
    //</editor-fold>//GEN-END:|440-getter|2|

    //<editor-fold defaultstate="collapsed" desc=" Generated Getter: okCommand_textBox_del
```

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

```
">//GEN-BEGIN:|441-getter|0|441-preInit
    /**
     * Returns an initiliazed instance of okCommand_textBox_del component.
     * @return the initialized component instance
     */
    public Command getOkCommand_textBox_del() {
        if (okCommand_textBox_del == null) {//GEN-END:|441-getter|0|441-preInit
            // write pre-init user code here
            okCommand_textBox_del = new Command("Ok", Command.OK, 0);
//GEN-LINE:|441-getter|1|441-postInit
            // write post-init user code here
        }//GEN-BEGIN:|441-getter|2|
        return okCommand_textBox_del;
    }
    //</editor-fold>//GEN-END:|441-getter|2|

    //<editor-fold defaultstate="collapsed" desc=" Generated Getter: backCommand_textBox_del
">//GEN-BEGIN:|445-getter|0|445-preInit
    /**
     * Returns an initiliazed instance of backCommand_textBox_del component.
     * @return the initialized component instance
     */
    public Command getBackCommand_textBox_del() {
        if (backCommand_textBox_del == null) {//GEN-END:|445-getter|0|445-preInit
            // write pre-init user code here
            backCommand_textBox_del = new Command("Back", Command.BACK, 0);
//GEN-LINE:|445-getter|1|445-postInit
            // write post-init user code here
        }//GEN-BEGIN:|445-getter|2|
        return backCommand_textBox_del;
    }
    //</editor-fold>//GEN-END:|445-getter|2|

    //<editor-fold defaultstate="collapsed" desc=" Generated Getter: choiceGroup_view
">//GEN-BEGIN:|448-getter|0|448-preInit
    /**
     * Returns an initiliazed instance of choiceGroup_view component.
     * @return the initialized component instance
     */
    public ChoiceGroup getChoiceGroup_view() {
        if (choiceGroup_view == null) {//GEN-END:|448-getter|0|448-preInit
            // write pre-init user code here
            /*
choiceGroup_view = new ChoiceGroup ("Registros disponibles:",
Choice.MULTIPLE);//GEN-LINE:|448-getter|1|448-postInit
            // write post-init user code here*/
            choiceGroup_view = generaChoiceGroup(vjson_view, choiceGroup_view);
        }//GEN-BEGIN:|448-getter|2|
        return choiceGroup_view;
    }
    //</editor-fold>//GEN-END:|448-getter|2|

    //<editor-fold defaultstate="collapsed" desc=" Generated Getter: textBox_edit
">//GEN-BEGIN:|449-getter|0|449-preInit
    /**
     * Returns an initiliazed instance of textBox_edit component.
     * @return the initialized component instance
```

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

```
     */
    public TextBox getTextBox_edit() {
        if (textBox_edit == null) {//GEN-END:|449-getter|0|449-preInit
            // write pre-init user code here

            //muestra los datos del registro seleccionado para editar
            arr_edit_mu5 = new JSONArray();
            arr_edit_mo5 = new JSONArray();
            arr_edit_fu5 = new JSONArray();
            arr_edit_fo5 = new JSONArray();
            try {
                json_edit = jObject[choiceGroup_edit.getSelectedIndex()];
                arr_edit_mu5 = json_edit.getJSONArray("mu5");
                arr_edit_mo5 = json_edit.getJSONArray("mo5");
                arr_edit_fu5 = json_edit.getJSONArray("fu5");
                arr_edit_fo5 = json_edit.getJSONArray("fo5");
                date_edit = json_edit.getString("date");

                //genera "str_edit": string que se representa en el textbox siguiente
                str_edit = "Date: "+date_edit+"\n";
                for (int i=0; i<4; i++) {
                    if (i == 0) {
                        str_edit += "male under 5:\n";
                        for (int j=0; j<arr_edit_mu5.length(); j++)
                            str_edit += arr_edit_mu5.getInt(j)+",";
                        str_edit = str_edit.substring(0, str_edit.length()-1);//quitar
última coma ,
                    }
                    else if (i == 1) {
                        str_edit += "male over 5:\n";
                        for (int k=0; k<arr_edit_mo5.length(); k++)
                            str_edit += arr_edit_mo5.getInt(k)+",";
                        str_edit = str_edit.substring(0, str_edit.length()-1);//quitar
última coma ,
                    }
                    else if (i == 2) {
                        str_edit += "female under 5:\n";
                        for (int l=0; l<arr_edit_fu5.length(); l++)
                            str_edit += arr_edit_fu5.getInt(l)+",";
                        str_edit = str_edit.substring(0, str_edit.length()-1);//quitar
última coma ,
                    }
                    else if (i == 3) {
                        str_edit += "female over 5:\n";
                        for (int m=0; m<arr_edit_fo5.length(); m++)
                            str_edit += arr_edit_fo5.getInt(m)+",";
                        str_edit = str_edit.substring(0, str_edit.length()-1);//quitar
última coma ,
                    }
                    str_edit += "\n";
                }
            }
            catch (JSONException jse) {
                jse.printStackTrace();
            }

            textBox_edit = new TextBox("textBox_edit", str_edit, 1000, TextField.ANY |
```

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

```
TextField.UNEDITABLE);//GEN-BEGIN:|449-getter|1|449-postInit
            textBox_edit.addCommand(getOkCommand_textBox_edit());
            textBox_edit.addCommand(getBackCommand_textBox_edit());
            textBox_edit.setCommandListener(this);//GEN-END:|449-getter|1|449-postInit
            // write post-init user code here
        }//GEN-BEGIN:|449-getter|2|
        return textBox_edit;
    }
    //</editor-fold>//GEN-END:|449-getter|2|

    //<editor-fold defaultstate="collapsed" desc=" Generated Getter: okCommand_textBox_edit
">//GEN-BEGIN:|451-getter|0|451-preInit
    /**
     * Returns an initiliazed instance of okCommand_textBox_edit component.
     * @return the initialized component instance
     */
    public Command getOkCommand_textBox_edit() {
        if (okCommand_textBox_edit == null) {//GEN-END:|451-getter|0|451-preInit
            // write pre-init user code here
            okCommand_textBox_edit = new Command("Ok", Command.OK, 0);
//GEN-LINE:|451-getter|1|451-postInit
            // write post-init user code here
        }//GEN-BEGIN:|451-getter|2|
        return okCommand_textBox_edit;
    }
    //</editor-fold>//GEN-END:|451-getter|2|

    //<editor-fold defaultstate="collapsed" desc=" Generated Getter:
backCommand_textBox_edit ">//GEN-BEGIN:|453-getter|0|453-preInit
    /**
     * Returns an initiliazed instance of backCommand_textBox_edit component.
     * @return the initialized component instance
     */
    public Command getBackCommand_textBox_edit() {
        if (backCommand_textBox_edit == null) {//GEN-END:|453-getter|0|453-preInit
            // write pre-init user code here
            backCommand_textBox_edit = new Command("Back", Command.BACK, 0);
//GEN-LINE:|453-getter|1|453-postInit
            // write post-init user code here
        }//GEN-BEGIN:|453-getter|2|
        return backCommand_textBox_edit;
    }
    //</editor-fold>//GEN-END:|453-getter|2|

    //<editor-fold defaultstate="collapsed" desc=" Generated Getter: choiceGroup_edit
">//GEN-BEGIN:|456-getter|0|456-preInit
    /**
     * Returns an initiliazed instance of choiceGroup_edit component.
     * @return the initialized component instance
     */
    public ChoiceGroup getChoiceGroup_edit() {
        if (choiceGroup_edit == null) {//GEN-END:|456-getter|0|456-preInit
            // write pre-init user code here
            /*
choiceGroup_edit = new ChoiceGroup ("Registros disponibles:",
Choice.MULTIPLE);//GEN-LINE:|456-getter|1|456-postInit
            // write post-init user code here*/
```

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

```
            choiceGroup_edit = generaChoiceGroup(vjson_edit, choiceGroup_edit);
        }//GEN-BEGIN:|456-getter|2|
        return choiceGroup_edit;
    }
    //</editor-fold>//GEN-END:|456-getter|2|

    //<editor-fold defaultstate="collapsed" desc=" Generated Getter:
form_edit_select_symptom ">//GEN-BEGIN:|457-getter|0|457-preInit
    /**
     * Returns an initiliazed instance of form_edit_select_symptom component.
     * @return the initialized component instance
     */
    public Form getForm_edit_select_symptom() {
        if (form_edit_select_symptom == null) {//GEN-END:|457-getter|0|457-preInit
            // write pre-init user code here
            form_edit_select_symptom = new Form("Editar registro", new Item[] {
getChoiceGroup_edit_symptom() });//GEN-BEGIN:|457-getter|1|457-postInit
            form_edit_select_symptom.addCommand(getOkCommand_edit_select_symptom());
            form_edit_select_symptom.addCommand(getBackCommand_edit_select_symptom());
            form_edit_select_symptom.setCommandListener(this);
//GEN-END:|457-getter|1|457-postInit
            // write post-init user code here
        }//GEN-BEGIN:|457-getter|2|
        return form_edit_select_symptom;
    }
    //</editor-fold>//GEN-END:|457-getter|2|

    //<editor-fold defaultstate="collapsed" desc=" Generated Getter:
okCommand_edit_select_symptom ">//GEN-BEGIN:|458-getter|0|458-preInit
    /**
     * Returns an initiliazed instance of okCommand_edit_select_symptom component.
     * @return the initialized component instance
     */
    public Command getOkCommand_edit_select_symptom() {
        if (okCommand_edit_select_symptom == null) {//GEN-END:|458-getter|0|458-preInit
            // write pre-init user code here
            okCommand_edit_select_symptom = new Command("Ok", Command.OK, 0);
//GEN-LINE:|458-getter|1|458-postInit
            // write post-init user code here
        }//GEN-BEGIN:|458-getter|2|
        return okCommand_edit_select_symptom;
    }
    //</editor-fold>//GEN-END:|458-getter|2|

    //<editor-fold defaultstate="collapsed" desc=" Generated Getter:
backCommand_edit_select_symptom ">//GEN-BEGIN:|460-getter|0|460-preInit
    /**
     * Returns an initiliazed instance of backCommand_edit_select_symptom component.
     * @return the initialized component instance
     */
    public Command getBackCommand_edit_select_symptom() {
        if (backCommand_edit_select_symptom == null) {//GEN-END:|460-getter|0|460-preInit
            // write pre-init user code here
            backCommand_edit_select_symptom = new Command("Back", Command.BACK, 0);
//GEN-LINE:|460-getter|1|460-postInit
            // write post-init user code here
        }//GEN-BEGIN:|460-getter|2|
```

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

```
        return backCommand_edit_select_symptom;
    }
    //</editor-fold>//GEN-END:|460-getter|2|

    //<editor-fold defaultstate="collapsed" desc=" Generated Getter:
choiceGroup_edit_symptom ">//GEN-BEGIN:|463-getter|0|463-preInit
    /**
     * Returns an initiliazed instance of choiceGroup_edit_symptom component.
     * @return the initialized component instance
     */
    public ChoiceGroup getChoiceGroup_edit_symptom() {
        if (choiceGroup_edit_symptom == null) {//GEN-END:|463-getter|0|463-preInit
            // write pre-init user code here
            //aquí generamos la lista de síntomas en función al fichero de configuración
            choiceGroup_edit_symptom = new ChoiceGroup("Seleccione s\u00EDntoma:", Choice.
EXCLUSIVE);
            for (int i=0;i<number_of_diseases;i++) {
                choiceGroup_edit_symptom.append(diseases[i], null);
            }
            for (int i=0;i<number_of_diseases;i++) {
                choiceGroup_edit_symptom.setFont(i, null);
            }
            choiceGroup_edit_symptom.setFitPolicy(Choice.TEXT_WRAP_DEFAULT);
            booleano = new boolean[number_of_diseases];
            for (int i=0;i<number_of_diseases;i++) {
                booleano[i] = false;
            }
            choiceGroup_edit_symptom.setSelectedFlags(booleano);
            /*
choiceGroup_edit_symptom = new ChoiceGroup ("Seleccione s\u00EDntoma:",
Choice.MULTIPLE);//GEN-LINE:|463-getter|1|463-postInit
            // write post-init user code here*/
        }//GEN-BEGIN:|463-getter|2|
        return choiceGroup_edit_symptom;
    }
    //</editor-fold>//GEN-END:|463-getter|2|

    //<editor-fold defaultstate="collapsed" desc=" Generated Getter: form_edit_number
">//GEN-BEGIN:|464-getter|0|464-preInit
    /**
     * Returns an initiliazed instance of form_edit_number component.
     * @return the initialized component instance
     */
    public Form getForm_edit_number() {
        if (form_edit_number == null) {//GEN-END:|464-getter|0|464-preInit
            // write pre-init user code here
            form_edit_number = new Form("Modifique el n\u00FAmero de casos", new Item[] {
getTextField_ed_mu5(), getTextField_ed_mo5(), getTextField_ed_fu5(), getTextField_ed_fo5()
});//GEN-BEGIN:|464-getter|1|464-postInit
            form_edit_number.addCommand(getOkCommand_edit_number());
            form_edit_number.addCommand(getBackCommand_edit_number());
            form_edit_number.setCommandListener(this);//GEN-END:|464-getter|1|464-postInit
            // write post-init user code here
        }//GEN-BEGIN:|464-getter|2|
        return form_edit_number;
    }
    //</editor-fold>//GEN-END:|464-getter|2|
```

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

```
    //<editor-fold defaultstate="collapsed" desc=" Generated Getter: textField_ed_mu5
">//GEN-BEGIN:|466-getter|0|466-preInit
    /**
     * Returns an initiliazed instance of textField_ed_mu5 component.
     * @return the initialized component instance
     */
    public TextField getTextField_ed_mu5() {
        if (textField_ed_mu5 == null) {//GEN-END:|466-getter|0|466-preInit
            // write pre-init user code here
            try {
                int mu5 = arr_edit_mu5.getInt(choiceGroup_edit_symptom.getSelectedIndex());
                if (mu5 != 0)
                    textField_ed_mu5 = new TextField("Male under 5 years", mu5+"", 32,
TextField.NUMERIC);//GEN-LINE:|466-getter|1|466-postInit
                else
                    textField_ed_mu5 = new TextField("Male under 5 years", "", 32, TextField.
NUMERIC);
                // write post-init user code here
            }
            catch ( JSONException jsone ) {
                jsone.printStackTrace();
            }
        }//GEN-BEGIN:|466-getter|2|
        return textField_ed_mu5;
    }
    //</editor-fold>//GEN-END:|466-getter|2|

    //<editor-fold defaultstate="collapsed" desc=" Generated Getter: textField_ed_mo5
">//GEN-BEGIN:|467-getter|0|467-preInit
    /**
     * Returns an initiliazed instance of textField_ed_mo5 component.
     * @return the initialized component instance
     */
    public TextField getTextField_ed_mo5() {
        if (textField_ed_mo5 == null) {//GEN-END:|467-getter|0|467-preInit
            // write pre-init user code here
            try {
                int mo5 = arr_edit_mo5.getInt(choiceGroup_edit_symptom.getSelectedIndex());
                if (mo5 != 0)
                    textField_ed_mo5 = new TextField("Male over 5 years", mo5+"", 32,
TextField.NUMERIC);//GEN-LINE:|467-getter|1|467-postInit
                else
                    textField_ed_mo5 = new TextField("Male over 5 years", "", 32, TextField.
NUMERIC);
                // write post-init user code here
            }
            catch ( JSONException jsone ) {
                jsone.printStackTrace();
            }
        }//GEN-BEGIN:|467-getter|2|
        return textField_ed_mo5;
    }
    //</editor-fold>//GEN-END:|467-getter|2|

    //<editor-fold defaultstate="collapsed" desc=" Generated Getter: textField_ed_fu5
">//GEN-BEGIN:|468-getter|0|468-preInit
```

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

```
    /**
     * Returns an initiliazed instance of textField_ed_fu5 component.
     * @return the initialized component instance
     */
    public TextField getTextField_ed_fu5() {
        if (textField_ed_fu5 == null) {//GEN-END:|468-getter|0|468-preInit
            // write pre-init user code here
            try {
                int fu5 = arr_edit_fu5.getInt(choiceGroup_edit_symptom.getSelectedIndex());
                if (fu5 != 0)
                    textField_ed_fu5 = new TextField("Female under 5 years", fu5+"", 32,
TextField.NUMERIC);//GEN-LINE:|468-getter|1|468-postInit
                else
                    textField_ed_fu5 = new TextField("Female under 5 years", "", 32,
TextField.NUMERIC);
                // write post-init user code here
            }
            catch ( JSONException jsone ) {
                jsone.printStackTrace();
            }
        }//GEN-BEGIN:|468-getter|2|
        return textField_ed_fu5;
    }
    //</editor-fold>//GEN-END:|468-getter|2|

    //<editor-fold defaultstate="collapsed" desc=" Generated Getter: textField_ed_fo5
">//GEN-BEGIN:|469-getter|0|469-preInit
    /**
     * Returns an initiliazed instance of textField_ed_fo5 component.
     * @return the initialized component instance
     */
    public TextField getTextField_ed_fo5() {
        if (textField_ed_fo5 == null) {//GEN-END:|469-getter|0|469-preInit
            // write pre-init user code here
            try {
                int fo5 = arr_edit_fo5.getInt(choiceGroup_edit_symptom.getSelectedIndex());
                if (fo5 != 0)
                    textField_ed_fo5 = new TextField("Female over 5 years", fo5+"", 32,
TextField.NUMERIC);//GEN-LINE:|469-getter|1|469-postInit
                else
                    textField_ed_fo5 = new TextField("Female over 5 years", "", 32, TextField
.NUMERIC);
                // write post-init user code here
            }
            catch ( JSONException jsone ) {
                jsone.printStackTrace();
            }
        }//GEN-BEGIN:|469-getter|2|
        return textField_ed_fo5;
    }
    //</editor-fold>//GEN-END:|469-getter|2|

    //<editor-fold defaultstate="collapsed" desc=" Generated Getter: bucle_edit
">//GEN-BEGIN:|474-getter|0|474-preInit
    /**
     * Returns an initiliazed instance of bucle_edit component.
     * @return the initialized component instance
```

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

```
     */
    public List getBucle_edit() {
        if (bucle_edit == null) {//GEN-END:|474-getter|0|474-preInit
            // write pre-init user code here
            bucle_edit = new List("list", Choice.IMPLICIT);
//GEN-BEGIN:|474-getter|1|474-postInit
            bucle_edit.append("Editar m\u00E1s s\u00EDntomas del mismo d\u00EDa", null);
            bucle_edit.append("Guardar y volver al Men\u00FA principal", null);
            bucle_edit.addCommand(getOkCommand_bucle_edit());
            bucle_edit.setCommandListener(this);
            bucle_edit.setSelectedFlags(new boolean[] { false, false });
//GEN-END:|474-getter|1|474-postInit
            // write post-init user code here
            //guardamos cambios realizados
            int sel_ind = choiceGroup_edit_symptom.getSelectedIndex();
            try {
                arr_edit_mu5.put(sel_ind, Integer.parseInt(textField_ed_mu5.getString()));
                arr_edit_mo5.put(sel_ind, Integer.parseInt(textField_ed_mo5.getString()));
                arr_edit_fu5.put(sel_ind, Integer.parseInt(textField_ed_fu5.getString()));
                arr_edit_fo5.put(sel_ind, Integer.parseInt(textField_ed_fo5.getString()));
            } catch (JSONException jsone) {
                jsone.printStackTrace();
            }
            try {
                jObject[choiceGroup_edit.getSelectedIndex()].put("mu5", arr_edit_mu5);
                jObject[choiceGroup_edit.getSelectedIndex()].put("mo5", arr_edit_mo5);
                jObject[choiceGroup_edit.getSelectedIndex()].put("fu5", arr_edit_fu5);
                jObject[choiceGroup_edit.getSelectedIndex()].put("fo5", arr_edit_fo5);
            } catch (JSONException jsone) {
                jsone.printStackTrace();
            }
        }//GEN-BEGIN:|474-getter|2|
        return bucle_edit;
    }
    //</editor-fold>//GEN-END:|474-getter|2|

    //<editor-fold defaultstate="collapsed" desc=" Generated Method: bucle_editAction
">//GEN-BEGIN:|474-action|0|474-preAction
    /**
     * Performs an action assigned to the selected list element in the bucle_edit component.
     */
    public void bucle_editAction() {//GEN-END:|474-action|0|474-preAction
        // enter pre-action user code here
        String __selectedString = getBucle_edit().getString(getBucle_edit().getSelectedIndex
());//GEN-BEGIN:|474-action|1|480-preAction
        if (__selectedString != null) {
            if (__selectedString.equals("Editar m\u00E1s s\u00EDntomas del mismo d\u00EDa"))
{//GEN-END:|474-action|1|480-preAction
                // write pre-action user code here
                resetForm_edit();//reseteo de algunos elementos del formulario
                switchDisplayable(null, getForm_edit_select_symptom());
//GEN-LINE:|474-action|2|480-postAction
                // write post-action user code here
            } else if (__selectedString.equals("Guardar y volver al Men\u00FA principal")) {
//GEN-LINE:|474-action|3|481-preAction
                // write pre-action user code here
                saveData sd_edit = new saveData();
```

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

```
                sd_edit.delete(f_rec);
                sd_edit.saveData(f_rec, jObject, false);
                vjson_del = vjson_edit = vjson_view = null;
                choiceGroup_del = choiceGroup_edit = choiceGroup_view = null;
                switchDisplayable(null, getMain());//GEN-LINE:|474-action|4|481-postAction
                // write post-action user code here
            }//GEN-BEGIN:|474-action|5|474-postAction
        }//GEN-END:|474-action|5|474-postAction
        // enter post-action user code here
    }//GEN-BEGIN:|474-action|6|
    //</editor-fold>//GEN-END:|474-action|6|

    //<editor-fold defaultstate="collapsed" desc=" Generated Getter: okCommand_edit_number
">//GEN-BEGIN:|470-getter|0|470-preInit
    /**
     * Returns an initiliazed instance of okCommand_edit_number component.
     * @return the initialized component instance
     */
    public Command getOkCommand_edit_number() {
        if (okCommand_edit_number == null) {//GEN-END:|470-getter|0|470-preInit
            // write pre-init user code here
            okCommand_edit_number = new Command("Ok", Command.OK, 0);
//GEN-LINE:|470-getter|1|470-postInit
            // write post-init user code here
        }//GEN-BEGIN:|470-getter|2|
        return okCommand_edit_number;
    }
    //</editor-fold>//GEN-END:|470-getter|2|

    //<editor-fold defaultstate="collapsed" desc=" Generated Getter: backCommand_edit_number
">//GEN-BEGIN:|472-getter|0|472-preInit
    /**
     * Returns an initiliazed instance of backCommand_edit_number component.
     * @return the initialized component instance
     */
    public Command getBackCommand_edit_number() {
        if (backCommand_edit_number == null) {//GEN-END:|472-getter|0|472-preInit
            // write pre-init user code here
            backCommand_edit_number = new Command("Back", Command.BACK, 0);
//GEN-LINE:|472-getter|1|472-postInit
            // write post-init user code here
        }//GEN-BEGIN:|472-getter|2|
        return backCommand_edit_number;
    }
    //</editor-fold>//GEN-END:|472-getter|2|

    //<editor-fold defaultstate="collapsed" desc=" Generated Getter: okCommand_bucle_edit
">//GEN-BEGIN:|478-getter|0|478-preInit
    /**
     * Returns an initiliazed instance of okCommand_bucle_edit component.
     * @return the initialized component instance
     */
    public Command getOkCommand_bucle_edit() {
        if (okCommand_bucle_edit == null) {//GEN-END:|478-getter|0|478-preInit
            // write pre-init user code here
            okCommand_bucle_edit = new Command("Ok", Command.OK, 0);
//GEN-LINE:|478-getter|1|478-postInit
```

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

```
            // write post-init user code here
        }//GEN-BEGIN:|478-getter|2|
        return okCommand_bucle_edit;
    }
    //</editor-fold>//GEN-END:|478-getter|2|

    //<editor-fold defaultstate="collapsed" desc=" Generated Getter: textBox_upload_HQ
">//GEN-BEGIN:|484-getter|0|484-preInit
    /**
     * Returns an initiliazed instance of textBox_upload_HQ component.
     * @return the initialized component instance
     */
    public TextBox getTextBox_upload_HQ() {
        if (textBox_upload_HQ == null) {//GEN-END:|484-getter|0|484-preInit
            // write pre-init user code here
            String str_ret_HQ = new String();
            for (int i = 0; i < n_recs; i++) {
                str_ret_HQ += return_HQ[i]+"\n";
            }
            textBox_upload_HQ = new TextBox("respuesta del HQ", str_ret_HQ, 10000, TextField.
ANY | TextField.UNEDITABLE);//GEN-BEGIN:|484-getter|1|484-postInit
            textBox_upload_HQ.addCommand(getOkCommand_upload_HQ());
            textBox_upload_HQ.setCommandListener(this);//GEN-END:|484-getter|1|484-postInit
            // write post-init user code here
        }//GEN-BEGIN:|484-getter|2|
        return textBox_upload_HQ;
    }
    //</editor-fold>//GEN-END:|484-getter|2|

    //<editor-fold defaultstate="collapsed" desc=" Generated Getter: okCommand_upload_HQ
">//GEN-BEGIN:|486-getter|0|486-preInit
    /**
     * Returns an initiliazed instance of okCommand_upload_HQ component.
     * @return the initialized component instance
     */
    public Command getOkCommand_upload_HQ() {
        if (okCommand_upload_HQ == null) {//GEN-END:|486-getter|0|486-preInit
            // write pre-init user code here
            okCommand_upload_HQ = new Command("Ok", Command.OK, 0);
//GEN-LINE:|486-getter|1|486-postInit
            // write post-init user code here
        }//GEN-BEGIN:|486-getter|2|
        return okCommand_upload_HQ;
    }
    //</editor-fold>//GEN-END:|486-getter|2|

    //<editor-fold defaultstate="collapsed" desc=" Generated Getter: okCommand_transfer
">//GEN-BEGIN:|489-getter|0|489-preInit
    /**
     * Returns an initiliazed instance of okCommand_transfer component.
     * @return the initialized component instance
     */
    public Command getOkCommand_transfer() {
        if (okCommand_transfer == null) {//GEN-END:|489-getter|0|489-preInit
            // write pre-init user code here
            okCommand_transfer = new Command("Ok", Command.OK, 0);
//GEN-LINE:|489-getter|1|489-postInit
```

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

```
            // write post-init user code here
        }//GEN-BEGIN:|489-getter|2|
        return okCommand_transfer;
    }
    //</editor-fold>//GEN-END:|489-getter|2|

    //<editor-fold defaultstate="collapsed" desc=" Generated Getter: waitScreen_HQ
">//GEN-BEGIN:|491-getter|0|491-preInit
    /**
     * Returns an initiliazed instance of waitScreen_HQ component.
     * @return the initialized component instance
     */
    public WaitScreen getWaitScreen_HQ() {
        if (waitScreen_HQ == null) {//GEN-END:|491-getter|0|491-preInit
            // write pre-init user code here
            waitScreen_HQ = new WaitScreen(getDisplay());
//GEN-BEGIN:|491-getter|1|491-postInit
            waitScreen_HQ.setTitle("waitScreen");
            waitScreen_HQ.setCommandListener(this);
            waitScreen_HQ.setText("Wait while uploading...");
            waitScreen_HQ.setTask(getTask_upload_HQ());//GEN-END:|491-getter|1|491-postInit
            // write post-init user code here
        }//GEN-BEGIN:|491-getter|2|
        return waitScreen_HQ;
    }
    //</editor-fold>//GEN-END:|491-getter|2|

    //<editor-fold defaultstate="collapsed" desc=" Generated Getter: task_upload_HQ
">//GEN-BEGIN:|494-getter|0|494-preInit
    /**
     * Returns an initiliazed instance of task_upload_HQ component.
     * @return the initialized component instance
     */
    public SimpleCancellableTask getTask_upload_HQ() {
        if (task_upload_HQ == null) {//GEN-END:|494-getter|0|494-preInit
            // write pre-init user code here
            task_upload_HQ = new SimpleCancellableTask();//GEN-BEGIN:|494-getter|1|494-execute
            task_upload_HQ.setExecutable(new org.netbeans.microedition.util.Executable() {
                public void execute() throws Exception {//GEN-END:|494-getter|1|494-execute
                    // write task_upload_HQ-execution user code here
                    conectaHTTP chttp;
                    String str_transfer = new String();
                    getJSON gj_transfer = new getJSON(f_rec);
                    n_recs = gj_transfer.getVjson().size();
                    Vector v_malos = new Vector();
                    return_HQ = new String[n_recs];
                    saveData sd_hq = new saveData();
                    for (int i = 0; i < n_recs; i++) {
                        str_transfer = gj_transfer.getVjson().elementAt(i).toString();
                        chttp = new conectaHTTP(str_transfer, urlserver, urlquery);
                        return_HQ[i] = chttp.retorno();
                        if (return_HQ[i].equals("OK")) {
                            sd_hq.saveData("sent_"+f_rec, gj_transfer.getVjson().elementAt(i
).toString(), false); //se copia el contenido viejo en un fichero
                        }
                        else {
                            v_malos.addElement(str_transfer);
```

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

```
                    }
                }
                saveData sd_del = new saveData();
                sd_del.saveData(f_rec, v_malos, true);
            }//GEN-BEGIN:|494-getter|2|494-postInit
        });//GEN-END:|494-getter|2|494-postInit
        // write post-init user code here
    }//GEN-BEGIN:|494-getter|3|
    return task_upload_HQ;
}
//</editor-fold>//GEN-END:|494-getter|3|

//<editor-fold defaultstate="collapsed" desc=" Generated Getter: textBox_upload_Base
">//GEN-BEGIN:|496-getter|0|496-preInit
/**
 * Returns an initiliazed instance of textBox_upload_Base component.
 * @return the initialized component instance
 */
public TextBox getTextBox_upload_Base() {
    if (textBox_upload_Base == null) {//GEN-END:|496-getter|0|496-preInit
        // write pre-init user code here
        textBox_upload_Base = new TextBox("textBox", null, 100, TextField.ANY);
//GEN-BEGIN:|496-getter|1|496-postInit
        textBox_upload_Base.addCommand(getOkCommand_upload_Base());
        textBox_upload_Base.setCommandListener(this);//GEN-END:|496-getter|1|496-postInit
        // write post-init user code here
    }//GEN-BEGIN:|496-getter|2|
    return textBox_upload_Base;
}
//</editor-fold>//GEN-END:|496-getter|2|

//<editor-fold defaultstate="collapsed" desc=" Generated Getter: textBox_upload_SD
">//GEN-BEGIN:|497-getter|0|497-preInit
/**
 * Returns an initiliazed instance of textBox_upload_SD component.
 * @return the initialized component instance
 */
public TextBox getTextBox_upload_SD() {
    if (textBox_upload_SD == null) {//GEN-END:|497-getter|0|497-preInit
        // write pre-init user code here
        textBox_upload_SD = new TextBox("textBox", null, 100, TextField.ANY);
//GEN-BEGIN:|497-getter|1|497-postInit
        textBox_upload_SD.addCommand(getOkCommand_upload_SD());
        textBox_upload_SD.setCommandListener(this);//GEN-END:|497-getter|1|497-postInit
        // write post-init user code here
    }//GEN-BEGIN:|497-getter|2|
    return textBox_upload_SD;
}
//</editor-fold>//GEN-END:|497-getter|2|

//<editor-fold defaultstate="collapsed" desc=" Generated Getter: waitScreen_Base
">//GEN-BEGIN:|504-getter|0|504-preInit
/**
 * Returns an initiliazed instance of waitScreen_Base component.
 * @return the initialized component instance
 */
public WaitScreen getWaitScreen_Base() {
```

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

```
        if (waitScreen_Base == null) {//GEN-END:|504-getter|0|504-preInit
            // write pre-init user code here
            waitScreen_Base = new WaitScreen(getDisplay());
//GEN-BEGIN:|504-getter|1|504-postInit
            waitScreen_Base.setTitle("waitScreen");
            waitScreen_Base.setCommandListener(this);
            waitScreen_Base.setText("Wait while transferring...");
            waitScreen_Base.setTask(getTask_upload_Base());
//GEN-END:|504-getter|1|504-postInit
            // write post-init user code here
        }//GEN-BEGIN:|504-getter|2|
        return waitScreen_Base;
    }
    //</editor-fold>//GEN-END:|504-getter|2|

    //<editor-fold defaultstate="collapsed" desc=" Generated Getter: waitScreen_SD
">//GEN-BEGIN:|508-getter|0|508-preInit
    /**
     * Returns an initiliazed instance of waitScreen_SD component.
     * @return the initialized component instance
     */
    public WaitScreen getWaitScreen_SD() {
        if (waitScreen_SD == null) {//GEN-END:|508-getter|0|508-preInit
            // write pre-init user code here
            waitScreen_SD = new WaitScreen(getDisplay());
//GEN-BEGIN:|508-getter|1|508-postInit
            waitScreen_SD.setTitle("waitScreen");
            waitScreen_SD.setCommandListener(this);
            waitScreen_SD.setText("Wait while copying...");
            waitScreen_SD.setTask(getTask_upload_SD());//GEN-END:|508-getter|1|508-postInit
            // write post-init user code here
        }//GEN-BEGIN:|508-getter|2|
        return waitScreen_SD;
    }
    //</editor-fold>//GEN-END:|508-getter|2|

    //<editor-fold defaultstate="collapsed" desc=" Generated Getter: okCommand_upload_Base
">//GEN-BEGIN:|498-getter|0|498-preInit
    /**
     * Returns an initiliazed instance of okCommand_upload_Base component.
     * @return the initialized component instance
     */
    public Command getOkCommand_upload_Base() {
        if (okCommand_upload_Base == null) {//GEN-END:|498-getter|0|498-preInit
            // write pre-init user code here
            okCommand_upload_Base = new Command("Ok", Command.OK, 0);
//GEN-LINE:|498-getter|1|498-postInit
            // write post-init user code here
        }//GEN-BEGIN:|498-getter|2|
        return okCommand_upload_Base;
    }
    //</editor-fold>//GEN-END:|498-getter|2|

    //<editor-fold defaultstate="collapsed" desc=" Generated Getter: okCommand_upload_SD
">//GEN-BEGIN:|500-getter|0|500-preInit
    /**
     * Returns an initiliazed instance of okCommand_upload_SD component.
```

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

```
     * @return the initialized component instance
     */
    public Command getOkCommand_upload_SD() {
        if (okCommand_upload_SD == null) {//GEN-END:|500-getter|0|500-preInit
            // write pre-init user code here
            okCommand_upload_SD = new Command("Ok", Command.OK, 0);
//GEN-LINE:|500-getter|1|500-postInit
            // write post-init user code here
        }//GEN-BEGIN:|500-getter|2|
        return okCommand_upload_SD;
    }
    //</editor-fold>//GEN-END:|500-getter|2|

    //<editor-fold defaultstate="collapsed" desc=" Generated Getter: task_upload_Base
">//GEN-BEGIN:|507-getter|0|507-preInit
    /**
     * Returns an initiliazed instance of task_upload_Base component.
     * @return the initialized component instance
     */
    public SimpleCancellableTask getTask_upload_Base() {
        if (task_upload_Base == null) {//GEN-END:|507-getter|0|507-preInit
            // write pre-init user code here
            task_upload_Base = new SimpleCancellableTask();
//GEN-BEGIN:|507-getter|1|507-execute
            task_upload_Base.setExecutable(new org.netbeans.microedition.util.Executable() {
                public void execute() throws Exception {//GEN-END:|507-getter|1|507-execute
                    // write task_upload_HQ-execution user code here
                    conectaHTTP chttp;
                    String str_transfer = new String();
                    getJSON gj_transfer = new getJSON(f_rec);
                    n_recs = gj_transfer.getVjson().size();
                    return_HQ = new String[n_recs];
                    for (int i = 0; i < n_recs; i++) {
                        str_transfer = gj_transfer.getVjson().elementAt(i).toString();
                        chttp = new conectaHTTP(str_transfer, urlserver, urlquery);
                        return_HQ[i] = chttp.retorno();
                    }
                }//GEN-BEGIN:|507-getter|2|507-postInit
            });//GEN-END:|507-getter|2|507-postInit
            // write post-init user code here
        }//GEN-BEGIN:|507-getter|3|
        return task_upload_Base;
    }
    //</editor-fold>//GEN-END:|507-getter|3|

    //<editor-fold defaultstate="collapsed" desc=" Generated Getter: task_upload_SD
">//GEN-BEGIN:|511-getter|0|511-preInit
    /**
     * Returns an initiliazed instance of task_upload_SD component.
     * @return the initialized component instance
     */
    public SimpleCancellableTask getTask_upload_SD() {
        if (task_upload_SD == null) {//GEN-END:|511-getter|0|511-preInit
            // write pre-init user code here
            task_upload_SD = new SimpleCancellableTask();//GEN-BEGIN:|511-getter|1|511-execute
            task_upload_SD.setExecutable(new org.netbeans.microedition.util.Executable() {
                public void execute() throws Exception {//GEN-END:|511-getter|1|511-execute
```

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

```
                    // write task_upload_HQ-execution user code here
                    conectaHTTP chttp;
                    String str_transfer = new String();
                    getJSON gj_transfer = new getJSON(f_rec);
                    n_recs = gj_transfer.getVjson().size();
                    return_SD = new String[n_recs];
                    for (int i = 0; i < n_recs; i++) {
                        str_transfer = gj_transfer.getVjson().elementAt(i).toString();
                        chttp = new conectaHTTP(str_transfer, urlserver, urlquery);
                        return_SD[i] = chttp.retorno();
                    }
                    saveData sd_sd = new saveData();
                    sd_sd.saveData("old"+f_rec, f_rec.toString(), false); //se copia el
contenido viejo en un fichero
                    sd_sd.saveData(f_rec, f_rec.toString(), true, ruta);  //se copia el
fichero en la SD
                    sd_sd.saveData(f_rec, "", true);                      //se borra el
contenido del fichero
                }//GEN-BEGIN:|511-getter|2|511-postInit
            });//GEN-END:|511-getter|2|511-postInit
            // write post-init user code here
        }//GEN-BEGIN:|511-getter|3|
        return task_upload_SD;
    }
    //</editor-fold>//GEN-END:|511-getter|3|


    //</editor-fold>
    //</editor-fold>

    //<editor-fold defaultstate="collapsed" desc=" Generated Getter: task_edit
">//GEN-BEGIN:|532-getter|0|532-preInit
    /**
     * Returns an initiliazed instance of task_edit component.
     * @return the initialized component instance
     */
    public SimpleCancellableTask getTask_edit() {
        if (task_edit == null) {//GEN-END:|532-getter|0|532-preInit
            // write pre-init user code here
            task_edit = new SimpleCancellableTask();//GEN-BEGIN:|532-getter|1|532-execute
            task_edit.setExecutable(new org.netbeans.microedition.util.Executable() {
                public void execute() throws Exception {//GEN-END:|532-getter|1|532-execute
                    // write task_edit-execution user code here
                    //guardar cambios realizados en fichero records.txt

                }//GEN-BEGIN:|532-getter|2|532-postInit
            });//GEN-END:|532-getter|2|532-postInit
            // write post-init user code here
        }//GEN-BEGIN:|532-getter|3|
        return task_edit;
    }
    //</editor-fold>//GEN-END:|532-getter|3|



    /**
     * Resetea los formularios form_add_select_symptom, form_add_number y sus componentes
```

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

```
     */
    public void resetForm_add () {
        /*if (form_add_select_symptom != null) {
            form_add_select_symptom.deleteAll();
            form_add_select_symptom.removeCommand(okCommand_select_symptom);
            form_add_select_symptom.removeCommand(backCommand_select_symptom);
            form_add_select_symptom = null;                 //apunta a null para que se
vuelva a generar
        }
        if (form_add_number != null) {
            form_add_number.deleteAll();
            form_add_number.removeCommand(okCommand_number);
            form_add_number.removeCommand(backCommand_number);
            form_add_number = null;
        }*/
        textBox_add = null;
        if (textField_mu5 != null)
            textField_mu5.setString("");
        if (textField_mo5 != null)
            textField_mo5.setString("");
        if (textField_fu5 != null)
            textField_fu5.setString("");
        if (textField_fo5 != null)
            textField_fo5.setString("");
        veamos = "";
    }

    /**
     * Resetea el formulario form_del y sus componentes
     */
    public void resetForm_del () {
        if (form_del != null) {
            form_del.deleteAll();
            form_del.removeCommand(okCommand_del);
            form_del.removeCommand(backCommand_del);
            form_del = null;
        }
        if (choiceGroup_del != null) {
            choiceGroup_del.deleteAll();
            choiceGroup_del = null;
        }
        textBox_del = null;
        str_del = "";
        //vjson_del.removeAllElements();
        //jObject = null;
    }

    /**
     * Resetea el formulario form_edit y sus componentes
     */
    public void resetForm_edit () {
        if (textBox_edit != null) {
            textBox_edit = null;
        }
        if (textField_ed_mu5 != null)
            textField_ed_mu5.setString("");
        if (textField_ed_mo5 != null)
```

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

```
            textField_ed_mo5.setString("");
        if (textField_ed_fu5 != null)
            textField_ed_fu5.setString("");
        if (textField_ed_fo5 != null)
            textField_ed_fo5.setString("");
        str_edit = "";
    }

    /**
     * Genera un ChoiceGroup con las fechas de los registros almacenados en el fichero de
texto
     * @param vjson es el vector con el que se genera el ChoiceGroup
     * @param choiceGroup es el nombre del ChoiceGroup sin elementos
     * @return devuelve el ChoiceGroup con los elementos
     */
    public ChoiceGroup generaChoiceGroup (Vector vjson, ChoiceGroup choiceGroup) {
        if (vjson == null) {
            getJSON gj = new getJSON(f_rec);    //lee el fichero de registros
            vjson = new Vector();
            vjson = gj.getVjson();              //mete los registros en el vector
        }
        int n_jsons = vjson.size();         //número de jsons en el fichero

        //se inicializa el array de objetos JSON con los json almacenados
        jObject = new JSONObject[n_jsons];//array de JSONObject del tamaño n_jsons
        try {
            for (int i=0; i<n_jsons; i++) {
                jObject[i] = new JSONObject(vjson.elementAt(i).toString());//se crea objeto
json con cada registro
            }
        } catch (JSONException jse) { jse.printStackTrace(); }

        choiceGroup = new ChoiceGroup("Registros disponibles:", Choice.EXCLUSIVE);
        try {
            for (int i=0; i<n_jsons; i++) {
                choiceGroup.append(jObject[i].getString("date"), null);//se muestra sólo la
fecha del json
            }
        } catch (JSONException jse) {
            jse.printStackTrace();
            choiceGroup.append("excepcion", null);
        }
        for (int i=0; i<n_jsons; i++) {
            choiceGroup.setFont(i, null);
        }
        choiceGroup.setFitPolicy(Choice.EXCLUSIVE);
        boolean[] bool = new boolean[n_jsons];
        for (int i=0; i<n_jsons; i++) {
            bool[i] = false;
        }
        choiceGroup.setSelectedFlags(bool);

        vjson_view = vjson_del = vjson_edit = vjson;

        return choiceGroup;
    }
```

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

```
    /**
     * Genera el String JSON
     * @return devuelve el JSON en un objeto String
     */
    public String generaJSON () {
        //formato:
{"operation":"id_operation","base":"id_base","device":"id_device","date":"20093112", ... }
        stringjson = "{\"operation\":\""+id_operation+"\",\"base\":\""+id_base+
"\",\"device\":\""+id_device+"\",\"date\":\""+date_add+"\",";
        stringjson += "\"mu5\":[";
        for (int i=0;i<number_of_diseases;i++) {
            stringjson += data[0][i]+",";
        }
        stringjson = stringjson.substring(0, stringjson.length()-1);//quitar última coma ,
        stringjson += "],";
        stringjson += "\"mo5\":[";
        for (int i=0;i<number_of_diseases;i++) {
            stringjson += data[1][i]+",";
        }
        stringjson = stringjson.substring(0, stringjson.length()-1);//quitar última coma ,
        stringjson += "],";
        stringjson += "\"fu5\":[";
        for (int i=0;i<number_of_diseases;i++) {
            stringjson += data[2][i]+",";
        }
        stringjson = stringjson.substring(0, stringjson.length()-1);//quitar última coma ,
        stringjson += "],";
        stringjson += "\"fo5\":[";
        for (int i=0;i<number_of_diseases;i++) {
            stringjson += data[3][i]+",";
        }
        stringjson = stringjson.substring(0, stringjson.length()-1);//quitar última coma ,
        stringjson += "]}";//cierra array y objeto
        return stringjson;
    }

    /**
     * Genera un String en formato yyyyMMdd a partir del objeto fecha
     * @param fecha objeto con el que se genera la fecha en String
     * @return devuelve el String en formato yyyyMMdd
     */
    public String generaDate (Date fecha) {
        String date;
        long date_s = fecha.getTime();
        Calendar c = Calendar.getInstance();
        c.setTime(new Date(date_s));
        int d = c.get(Calendar.DAY_OF_MONTH);
        int m = c.get(Calendar.MONTH)+1;//el primer mes de Calendar.MONTH es 0
        int y = c.get(Calendar.YEAR);
        if (m < 10)                     //generamos una fecha en formato yyyymmdd
            if (d < 10)
                date = y+"0"+m+"0"+d;
            else
                date = y+"0"+m+""+d;
        else
            if (d < 10)
                date = y+""+m+"0"+d;
```

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

```
            else
                date = y+""+m+""+d;
        return date;
    }

    /**
     * Returns a display instance.
     * @return the display instance.
     */
    public Display getDisplay () {
        return Display.getDisplay(this);
    }

    /**
     * Exits MIDlet.
     */
    public void exitMIDlet() {
        switchDisplayable (null, null);
        destroyApp(true);
        notifyDestroyed();
    }

    /**
     * Called when MIDlet is started.
     * Checks whether the MIDlet have been already started and initialize/starts or resumes
the MIDlet.
     */
    public void startApp() {
        if (midletPaused) {
            resumeMIDlet ();
        } else {
            initialize ();
            startMIDlet ();
        }
        midletPaused = false;
    }

    /**
     * Called when MIDlet is paused.
     */
    public void pauseApp() {
        midletPaused = true;
    }

    /**
     * Called to signal the MIDlet to terminate.
     * @param unconditional if true, then the MIDlet has to be unconditionally terminated
and all resources has to be released.
     */
    public void destroyApp(boolean unconditional) {
    }

}
```

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

```
/**
 * myEpideMobile. Aplicación de Gestión de información epidemiológica en emergencias
internacionales.
 *
 * Copyright (C) 2010 Daniel Mayor Azcona daniel.mayor@unavarra.es
 *
 * Este programa es software libre. Puede redistribuirlo y/o modificarlo bajo los términos
de la Licencia
 * Pública General de GNU según es publicada por la Free Software Foundation, bien de la
versión 2 de dicha
 * Licencia o bien de cualquier versión posterior.
 *
 * Este programa se distribuye con la esperanza de que sea útil, pero SIN NINGUNA GARANTÍA,
incluso sin la
 * garantía MERCANTIL implícita o sin garantizar la CONVENIENCIA PARA UN PROPÓSITO
PARTICULAR. Véase la Licencia
 * Pública General de GNU para más detalles.
 *
 * Debería haber recibido una copia de la Licencia Pública General junto con este programa.
Si no ha sido así,
 * escriba a la Free Software Foundation, Inc., en 675 Mass Ave, Cambridge, MA 02139, EEUU.
 *
 */

import javax.microedition.io.Connector;
import javax.microedition.io.HttpConnection;
import java.io.DataInputStream;
import java.io.DataOutputStream;
import java.io.IOException;
import java.io.OutputStream;

/**
 * Realiza las conexiones con el clúster central mediante el protocolo HTTP.
 * @author Daniel
 */
public class conectaHTTP {

    String salida = "CONNECTION FAILURE";
    DataInputStream dis = null;

    /**
     * Constructor de la clase conectaHTTP.
     * @param stringjson texto que se transmite vía POST al servidor.
     * @param urlquery consulta que se envía al servidor en formato <var>=.
     * @param urlserver URL del servidor.
     */
    public conectaHTTP(String stringjson, String urlserver, String urlquery){
        try {
            this.run(stringjson, urlserver, urlquery);
        } catch (Exception e) { e.printStackTrace(); }
    }

    /**
     * Muestra la respuesta del servidor.
     * @return mensaje de retorno del servidor.
     */
    public String retorno() {
```

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

```
        return salida;
    }

    /**
     * Lanza la conexión contra el servidor del clúster y envía el json por el método POST
     * @param stringjson texto que se transmite vía POST al servidor.
     * @param urlquery consulta que se envía al servidor en formato <var>=.
     * @param urlserver URL del servidor.
     */
    private void run(String stringjson, String urlserver, String urlquery) throws IOException
{
        String url = urlserver;
        HttpConnection hc = null;
        OutputStream os = null;
        DataOutputStream dos = null;

        int rc;

        try {
            hc = (HttpConnection)Connector.open(url, Connector.READ_WRITE);
            hc.setRequestMethod(HttpConnection.POST);

            hc.setRequestProperty("Content-Type", "application/x-www-form-urlencoded");
            hc.setRequestProperty("User-Agent","Profile/MIDP-2.0 Configuration/CLDC-1.1");
            hc.setRequestProperty("Content-Language", "es-ES");

            dos = hc.openDataOutputStream();
            String consulta = urlquery+stringjson;

            //envía byte a byte la consulta vía POST
            byte[] byteRequest = consulta.getBytes();
            for (int i = 0; i < byteRequest.length; i++) {
                dos.writeByte(byteRequest[i]);
            }

            rc = hc.getResponseCode();
            if (rc != HttpConnection.HTTP_OK) {
                throw new IOException("HTTP response code: " + rc);
            }

            //lee la entrada del DataInputStream que se recibe del servidor
            dis = hc.openDataInputStream();
            StringBuffer sb = new StringBuffer();
            int b;
            while ((b = dis.read()) != -1) {
                if (b == 10)                //si el byte leído es un salto de línea,
cambiarlo por ;
                    sb.append("\n");
                else
                    sb.append((char)b);
            }
            salida = sb.toString();
        } catch (Exception e) { e.printStackTrace(); }
        finally {
            try {
                if (dis != null) dis.close();
                if (os != null) os.close();
```

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

```
                if (hc != null) hc.close();
            } catch (Exception e) { e.printStackTrace(); }
        }
    }
}
```

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

```
/**
 * myEpideMobile. Aplicación de Gestión de información epidemiológica en emergencias internacionales.
 *
 * Copyright (C) 2010 Daniel Mayor Azcona daniel.mayor@unavarra.es
 *
 * Este programa es software libre. Puede redistribuirlo y/o modificarlo bajo los términos de la Licencia
 * Pública General de GNU según es publicada por la Free Software Foundation, bien de la versión 2 de dicha
 * Licencia o bien de cualquier versión posterior.
 *
 * Este programa se distribuye con la esperanza de que sea útil, pero SIN NINGUNA GARANTÍA, incluso sin la
 * garantía MERCANTIL implícita o sin garantizar la CONVENIENCIA PARA UN PROPÓSITO PARTICULAR. Véase la Licencia
 * Pública General de GNU para más detalles.
 *
 * Debería haber recibido una copia de la Licencia Pública General junto con este programa. Si no ha sido así,
 * escriba a la Free Software Foundation, Inc., en 675 Mass Ave, Cambridge, MA 02139, EEUU.
 *
 */

import java.io.InputStream;
import org.json.me.JSONObject;
import java.util.Vector;

/**
 * Accede al fichero de configuración settings.txt y obtiene las variables globales con los ajustes del programa.
 * @author Daniel Mayor
 */
public class getConfigIDs {
    int n = 3;
    Vector ids = new Vector();
    StringBuffer sb;
    //String[] ids = new String[5];
    int i = 0, c;

    /**
     * Constructor de la clase getConfigIDs. Lee el fichero de configuración y almacena
     * las variables en un vector.
     * @param fichero nombre del fichero.
     */
    public getConfigIDs(String fichero) {
        InputStream is = null;
        sb = new StringBuffer();
        try {
            is = getClass().getResourceAsStream(fichero);
            while ((c = is.read()) != -1) {
                if (c == 10) { //almacena el string acumulado en una nueva posición del array con el LF
                    ids.addElement(sb.toString());
                    sb = null;
                    sb = new StringBuffer();
                    i++;
```

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

```
                }
                else if (c == 13) //quita el retorno de carro CR
                    {}
                else
                    sb.append((char)c);
            }
            if (c == -1)
                if (sb.length() > 1) {
                    ids.addElement(sb.toString());
                    sb = null;
                }
            is.close();
            is = null;
        } catch (Exception e) { e.printStackTrace(); }
    }

    /**
     * Devuelve un array de Strings con el contenido del vector ids
     * @return idstring contiene los elementos del vector
     */
    public String[] idstring() {
        String[] idstring = new String[ids.size()];
        for (int j=0; j<ids.size(); j++)
            idstring[j] = ids.elementAt(j).toString();
        return idstring;
    }

    /**
     * Convierte el vector con los json en un objeto JSON
     * @deprecated no se utiliza
     * @return jObject objeto JSON
     */
    public JSONObject vector2JSON() {
        JSONObject jObject = new JSONObject();//array de JSONObject del tamaño gj2.i
        try {
            for (int j=0; j<ids.size(); j++) {
                jObject.getJSONArray(sb.toString());
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
        return jObject;
    }
}
```

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

```
/**
 * myEpideMobile. Aplicación de Gestión de información epidemiológica en emergencias internacionales.
 *
 * Copyright (C) 2010 Daniel Mayor Azcona daniel.mayor@unavarra.es
 *
 * Este programa es software libre. Puede redistribuirlo y/o modificarlo bajo los términos de la Licencia
 * Pública General de GNU según es publicada por la Free Software Foundation, bien de la versión 2 de dicha
 * Licencia o bien de cualquier versión posterior.
 *
 * Este programa se distribuye con la esperanza de que sea útil, pero SIN NINGUNA GARANTÍA, incluso sin la
 * garantía MERCANTIL implícita o sin garantizar la CONVENIENCIA PARA UN PROPÓSITO PARTICULAR. Véase la Licencia
 * Pública General de GNU para más detalles.
 *
 * Debería haber recibido una copia de la Licencia Pública General junto con este programa. Si no ha sido así,
 * escriba a la Free Software Foundation, Inc., en 675 Mass Ave, Cambridge, MA 02139, EEUU.
 *
 */

import java.io.InputStream;
import java.util.Vector;

/**
 * Lee del fichero de configuración symptoms.txt los síntomas de enfermedades que se analizarán.
 * @author Daniel Mayor
 */
public class getConfigSymptom {
    int num_sintomas;
    String[] sintomas;//habría que afinar esto
    int c;
    Vector v = new Vector();

    /**
     * Constructor de la clase getConfigSymptom. Lee el fichero con los síntomas y los almacena en un vector.
     * @param fichero nombre del fichero con los síntomas.
     */
    public getConfigSymptom(String fichero) {
        InputStream is = null;
        StringBuffer sb = new StringBuffer();
        try {
            is = getClass().getResourceAsStream(fichero);
            while ((c = is.read()) != -1) {
                if (c == 10) { //almacena el String acumulado en una nueva posición del array con el LF
                    v.addElement(sb.toString());
                    sb = null;
                    sb = new StringBuffer();
                }
                else if (c == 13) //ignora el retorno de carro CR
                    {}
```

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

```
                else
                    sb.append((char)c);
            }
            if (c == -1)
                if (sb.length() > 1) {
                    v.addElement(sb.toString());
                    sb = null;
                }
            num_sintomas = v.size();
            sintomas = new String[v.size()];
            for(int i=0; i<v.size(); i++){
                sintomas[i]=(v.elementAt(i).toString());
            }
            is.close();
            is = null;
        } catch (Exception e) {
            e.printStackTrace();
        }
    }

    /**
     * Devuelve el número de síntomas.
     * @return num_sintomas número de síntomas.
     */
    public int n_sintomas() {
        return num_sintomas;
    }

    /**
     * Devuelve un array de Strings que almacena los síntomas.
     * @return sintomas array de Strings con los síntomas.
     */
    public String[] sintomas() {
        return sintomas;
    }
}
```

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

```
/**
 * myEpideMobile. Aplicación de Gestión de información epidemiológica en emergencias internacionales.
 *
 * Copyright (C) 2010 Daniel Mayor Azcona daniel.mayor@unavarra.es
 *
 * Este programa es software libre. Puede redistribuirlo y/o modificarlo bajo los términos de la Licencia
 * Pública General de GNU según es publicada por la Free Software Foundation, bien de la versión 2 de dicha
 * Licencia o bien de cualquier versión posterior.
 *
 * Este programa se distribuye con la esperanza de que sea útil, pero SIN NINGUNA GARANTÍA, incluso sin la
 * garantía MERCANTIL implícita o sin garantizar la CONVENIENCIA PARA UN PROPÓSITO PARTICULAR. Véase la Licencia
 * Pública General de GNU para más detalles.
 *
 * Debería haber recibido una copia de la Licencia Pública General junto con este programa. Si no ha sido así,
 * escriba a la Free Software Foundation, Inc., en 675 Mass Ave, Cambridge, MA 02139, EEUU.
 *
 */

import java.io.InputStream;
import java.io.IOException;
import javax.microedition.io.Connector;
import javax.microedition.io.file.FileConnection;
import java.util.Vector;
import org.json.me.JSONObject;
import org.json.me.JSONException;

/**
 * Lee los ficheros de texto almacenados en el dispositivo móvil con los datos recogidos por el sistema.
 * @author Daniel Mayor
 */
public class getJSON {
    int c = 0;
    Vector v = new Vector();
    int exito = 0;
    StringBuffer sb;
    InputStream in;
    FileConnection fc;

    /**
     * Constructor de la clase getJSON. Abre un fichero con objetos JSON, los ordena por fecha y los mete en un vector.
     * @param fichero nombre del fichero.
     */
    public getJSON(String fichero) {
        //String readFile = System.getProperty("fileconn.dir.memorycard")+fichero; //tarjeta de memoria
        String readFile = "file:///root1/"+fichero; //emulador NetBeans
        sb = new StringBuffer();
        try {
            fc = (FileConnection) Connector.open(readFile, Connector.READ);
```

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

```
            try {
                in = fc.openInputStream();
            } catch (IOException ioe) { ioe.printStackTrace(); }
            while ((c = in.read()) != -1) {
                if (c == 10) { //almacena el string acumulado en una nueva posición del
array cuando encuentra un LF
                    v.addElement(sb.toString());
                    sb = null;
                    sb = new StringBuffer();
                }
                else if (c == 13) {}//ignora el retorno de carro CR
                else
                    sb.append((char)c);
            }
            if (c == -1)
                if (sb.length() > 1) {
                    v.addElement(sb.toString());
                    sb = null;
                }
            fc.close();
        } catch (IOException ioe) {
            ioe.printStackTrace();
            exito = -1;
        } catch (NullPointerException npe) {
            npe.printStackTrace();
            exito = -2;
        } catch (Exception e) {
            e.printStackTrace();
            exito = -3;
        }
        v.trimToSize();
        v = sortVector(v, "date");
        exito = 1;
    }

    /**
     * Constructor de la clase getJSON. Abre un fichero con objetos JSON, los ordena por
fecha y los mete en un vector.
     * @param fichero nombre del fichero.
     * @param ruta ruta del directorio del fichero que se va a leer.
     */
    public getJSON(String fichero, String ruta) {
        //String readFile = System.getProperty("fileconn.dir.memorycard")+fichero;
//tarjeta de memoria
        String readFile = "file:///root1/"+fichero; //emulador NetBeans
        sb = new StringBuffer();
        try {
            fc = (FileConnection) Connector.open(readFile, Connector.READ);
            try {
                in = fc.openInputStream();
            } catch (IOException ioe) { ioe.printStackTrace(); }
            while ((c = in.read()) != -1) {
                if (c == 10) { //almacena el string acumulado en una nueva posición del
array cuando encuentra un LF
                    v.addElement(sb.toString());
                    sb = null;
                    sb = new StringBuffer();
```

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

```
                }
                else if (c == 13) {}    //ignora el retorno de carro CR
                else
                    sb.append((char)c);
            }
            if (c == -1)
                if (sb.length() > 1) {
                    v.addElement(sb.toString());
                    sb = null;
                }
            fc.close();
        } catch (IOException ioe) {
            ioe.printStackTrace();
            exito = -1;
        } catch (NullPointerException npe) {
            npe.printStackTrace();
            exito = -2;
        } catch (Exception e) {
            e.printStackTrace();
            exito = -3;
        }
        v.trimToSize();
        v = sortVector(v, "date");
        exito = 1;
    }

    /**
     * Devuelve el vector con los registros de los json almacenados.
     * @return v vector con los registros de los json almacenados.
     */
    public Vector getVjson() {
        return v;
    }

    /**
     * Convierte el vector v en un array de Strings.
     * @return jsons array de Strings en el que almacena los json almacenados en el vector.
     */
    public String[] jsons() {
        String[] jsons = new String[v.size()];
        for(int i=0; i<v.size(); i++){
            jsons[i] = v.elementAt(i).toString();
        }
        return jsons;
    }

    /**
     * Devuelve el tamaño del vector.
     * @return v.size() tamaño del vector.
     */
    public int n_jsons() {
        return v.size();
    }

    /**
     * Devuelve un los elementos del vector concatenados y separados por un salto de línea
     * @return cont String con los elementos del vector
```

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

```
     */
    public String all() {
        String cont;
        if (v.size() > 0) {
            cont = v.elementAt(0).toString();
            for(int i = 1; i < v.size(); i++) {
                cont += v.elementAt(i).toString()+"\n";
            }
        }
        else
            cont = "Fichero vacío";
        return cont;
    }

    /**
     * Ordena los registros de los json almacenados en el vector de menor a mayor
     * en función del key introducido con el algoritmo de la burbuja.
     * @param key nombre de la variable que se utiliza para realizar la ordenación.
     * @param v vector que se va a ordenar.
     * @return v_sorted vector ordenado.
     */
    public Vector sortVector(Vector v, String key) {
        JSONObject[] jsones = new JSONObject[v.size()];
        String[] dates = new String[v.size()];          //guarda las fechas
        Vector v_sorted = new Vector();
        try {
            for(int i = 0; i < v.size(); i++){
                jsones[i] = new JSONObject(v.elementAt(i).toString());
                dates[i] = jsones[i].get("date").toString();  //almacena la fecha
            }
        } catch (JSONException jse) {
            jse.printStackTrace();
        }
        ordenamientoBurbuja(dates, v.size());
        //ahora que están las fechas ordenadas
        //se va a generar el nuevo vector con las fechas ordenadas
        int j;
        try {
            for (int i = 0; i < v.size(); i++){
                j = 0;
                while (dates[i].compareTo(jsones[j].get("date").toString()) != 0) {
                    j++;
                }
                v_sorted.addElement(jsones[j]);
            }
        } catch (JSONException jse) {
            jse.printStackTrace();
        }
        v_sorted.trimToSize();
        return v_sorted;
    }

    /**
     * Algoritmo de ordenación por el método de la burbuja.
     * @param util_v número de elementos del conjunto que se va a ordenar.
     * @param v elementos que se van a ordenar.
```

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

```
    public void ordenamientoBurbuja(String v[], int util_v) {
        String temp;
        for (int i = 0; i <= util_v - 2; i++) {
            for (int j = i + 1; j <= util_v - 1; j++) {
                if (v[i].compareTo(v[j]) > 0) {
                    temp = v[i];
                    v[i] = v[j];
                    v[j] = temp;
                }
            }
        }
    }

}
```

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

```
/**
 * myEpideMobile. Aplicación de Gestión de información epidemiológica en emergencias internacionales.
 *
 * Copyright (C) 2010 Daniel Mayor Azcona daniel.mayor@unavarra.es
 *
 * Este programa es software libre. Puede redistribuirlo y/o modificarlo bajo los términos de la Licencia
 * Pública General de GNU según es publicada por la Free Software Foundation, bien de la versión 2 de dicha
 * Licencia o bien de cualquier versión posterior.
 *
 * Este programa se distribuye con la esperanza de que sea útil, pero SIN NINGUNA GARANTÍA, incluso sin la
 * garantía MERCANTIL implícita o sin garantizar la CONVENIENCIA PARA UN PROPÓSITO PARTICULAR. Véase la Licencia
 * Pública General de GNU para más detalles.
 *
 * Debería haber recibido una copia de la Licencia Pública General junto con este programa. Si no ha sido así,
 * escriba a la Free Software Foundation, Inc., en 675 Mass Ave, Cambridge, MA 02139, EEUU.
 *
 */

import java.io.InputStream;
import java.util.Vector;

/**
 * Lee los ficheros de texto almacenados en el archivo .jar del dispositivo móvil con los datos
 * recogidos por el sistema.
 * @author Daniel Mayor
 */
public class getJSON_jar {
    int c;
    Vector v = new Vector();

    /**
     * Constructor de la clase getJSON_jar. Lee los ficheros de texto almacenados dentro del .jar.
     * @param fichero nombre del fichero.
     */
    public getJSON_jar(String fichero) {
        InputStream is = null;
        StringBuffer sb = new StringBuffer();
        try {
            is = getClass().getResourceAsStream(fichero);
            while ((c = is.read()) != -1) {
                if (c == 10) { //almacena el string acumulado en una nueva posición del array con el LF
                    v.addElement(sb.toString());
                    sb = null;
                    sb = new StringBuffer();
                }
                else if (c == 13) //quita el retorno de carro CR
                    {}
                else
```

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

```
                    sb.append((char)c);
            }
            if (c == -1)
                if (sb.length() > 1) {
                    v.addElement(sb.toString());
                    sb = null;
                }
            is.close();
            is = null;
        } catch (Exception e) { e.printStackTrace(); }
    }

    /**
     * Devuelve el vector en forma de array de Strings.
     * @return jsons array de Strings con los JSON almacenados en el vector.
     */
    public String[] jsons() {
        String[] jsons = new String[v.size()];
        for(int i=0; i<v.size(); i++){
            jsons[i]=(v.elementAt(i).toString());
        }
        return jsons;
    }

    /**
     * Devuelve el tamaño del vector.
     * @return v.size() número de json almacenados en el fichero
     */
    public int n_jsons() {
        return v.size();
    }
}
```

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

```
/**
 * myEpideMobile. Aplicación de Gestión de información epidemiológica en emergencias internacionales.
 *
 * Copyright (C) 2010 Daniel Mayor Azcona daniel.mayor@unavarra.es
 *
 * Este programa es software libre. Puede redistribuirlo y/o modificarlo bajo los términos de la Licencia
 * Pública General de GNU según es publicada por la Free Software Foundation, bien de la versión 2 de dicha
 * Licencia o bien de cualquier versión posterior.
 *
 * Este programa se distribuye con la esperanza de que sea útil, pero SIN NINGUNA GARANTÍA, incluso sin la
 * garantía MERCANTIL implícita o sin garantizar la CONVENIENCIA PARA UN PROPÓSITO PARTICULAR. Véase la Licencia
 * Pública General de GNU para más detalles.
 *
 * Debería haber recibido una copia de la Licencia Pública General junto con este programa. Si no ha sido así,
 * escriba a la Free Software Foundation, Inc., en 675 Mass Ave, Cambridge, MA 02139, EEUU.
 *
 */

import java.io.IOException;
import java.io.OutputStream;
import java.io.PrintStream;
import javax.microedition.io.Connector;
import javax.microedition.io.file.FileConnection;
import java.util.Vector;
import org.json.me.JSONObject;

/**
 * Contiene métodos para guardar los datos recogidos con el programa en ficheros de texto almacenados
 * en el dispositivo móvil.
 * @author Daniel Mayor
 */
public class saveData {
    int exito = 0;

    /**
     * Constructor de la clase saveData. Guarda un json en forma de String en un fichero de texto.
     * Tiene la opción de sobrescribir el fichero o escribir al final de éste.
     * @param fichero nombre del fichero donde se va a guardar la información.
     * @param json objeto que se va a guardar en el fichero.
     * @param sobreescribir si es true se sobreescribirá el contenido fichero. Si es false se respetará el contenido
     * y la información se añadirá al final del fichero.
     * @return código de retorno del constructor. Si finaliza con éxito devuelve 1.
     */
    public int saveData(String fichero, String json, final boolean sobreescribir) {
        final String writeFile, writeContents;
        try {
            //emulador NetBeans file:///root1/
            //writeFile = "file:///C:/"+fichero;
```

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

```
            //writeFile = System.getProperty("fileconn.dir.memorycard")+fichero;
            writeFile = "file:///root1/"+fichero;
            writeContents = json;
            try {
                OutputStream os;
                FileConnection fc = (FileConnection) Connector.open(writeFile, Connector.
READ_WRITE);
                if (!fc.exists())
                    fc.create();
                if (sobreescribir) {
                    os = fc.openOutputStream();//empieza a escribir al principio del fichero
                }
                else
                    os = fc.openOutputStream(fc.fileSize());//le decimos que empiece al final
                PrintStream ps = new PrintStream(os);
                ps.println(writeContents);
                os.flush();
                os.close();
                fc.close();
                exito = 1;
            } catch (IOException e){
                e.printStackTrace();
                exito = -1;
            } catch (SecurityException e){
                e.printStackTrace();
                exito = -2;
            }
        }
        catch (Exception e) {
            e.printStackTrace();
            exito = -1;
        }
        return exito;
    }

    /**
     * Constructor de la clase saveData. Guarda un vector de json en forma de String en un
fichero de texto.
     * Tiene la opción de sobrescribir el fichero o escribir al final de éste.
     * @param fichero nombre del fichero donde se va a guardar la información.
     * @param sobreescribir si es true se sobreescribirá el contenido fichero. Si es false
se respetará el contenido
     * y la información se añadirá al final del fichero.
     * @param vjson vector cuyos elementos serán guardados en el fichero separados por
saltos de línea.
     * @return código de retorno del constructor. Si finaliza con éxito devuelve 1.
     */
    public int saveData(String fichero, Vector vjson, final boolean sobreescribir) {
        final String writeFile;
        //emulador NetBeans file:///root1/
        //writeFile = "file:///C:/"+fichero;
        //writeFile = System.getProperty("fileconn.dir.memorycard")+fichero;
        writeFile = "file:///root1/"+fichero;
        try {
            OutputStream os;
            FileConnection fc = (FileConnection) Connector.open(writeFile, Connector.
READ_WRITE);
```

upna Universidad Pública de Navarro Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

```
            if (!fc.exists())
                fc.create();
            if (sobreescribir) {
                fc.truncate(0);//elimina el contenido del fichero registros
                os = fc.openOutputStream();
                vjson.trimToSize();
                fc.fileSize();
            }
            else
                os = fc.openOutputStream(fc.fileSize());//para que empiece a escribir al
final del fichero
            PrintStream ps = new PrintStream(os);
            for (int i=0; i<vjson.size(); i++)
                ps.println(vjson.elementAt(i).toString());
            fc.close();
            exito = 1;
        }
        catch (IOException e) {
            e.printStackTrace();
            exito = -1;
        }
        catch (SecurityException se) {
            se.printStackTrace();
            exito = -2;
        }
        return exito;
    }

    /**
     * Constructor de la clase saveData. Guarda un array de objetos JSON en un fichero de
texto.
     * Tiene la opción de sobrescribir el fichero o escribir al final de éste.
     * @param fichero nombre del fichero donde se va a guardar la información.
     * @param jObject array de objetos JSON cuyos elementos serán guardados en el fichero
separados por saltos de línea.
     * @param sobreescribir si es true se sobreescribirá el contenido fichero. Si es false
se respetará el contenido
     * y la información se añadirá al final del fichero.
     * @return código de retorno del constructor. Si finaliza con éxito devuelve 1.
     */
    public int saveData(String fichero, JSONObject[] jObject, final boolean sobreescribir) {
        final String writeFile;
        Vector v = new Vector();
        for (int i=0; i<jObject.length; i++) {
            v.addElement(jObject[i].toString());
        }
        //emulador NetBeans file:///root1/
        //writeFile = "file:///C:/"+fichero;
        //writeFile = System.getProperty("fileconn.dir.memorycard")+fichero;
        writeFile = "file:///root1/"+fichero;
        try {
            OutputStream os;
            //FileConnection fc = (FileConnection) Connector.open(writeFile,
Connector.READ_WRITE);
            FileConnection fc = (FileConnection) Connector.open(writeFile, Connector.
READ_WRITE);
            if (!fc.exists())
```

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

```
                fc.create();
            if (sobreescribir)
                os = fc.openOutputStream();
            else
                os = fc.openOutputStream(fc.fileSize());//empieza a escribir al final del
fichero
            PrintStream ps = new PrintStream(os);
            for (int i=0; i<v.size(); i++)
                ps.println(v.elementAt(i).toString());
            fc.close();
            exito = 1;
        }
        catch (IOException e){
            e.printStackTrace();
            exito = -1;
        }
        return exito;
    }

    /**
     * Constructor de la clase saveData. Guarda un json en forma de String en un fichero de
texto.
     * Tiene la opción de sobrescribir el fichero o escribir al final de éste.
     * @param fichero nombre del fichero donde se va a guardar la información.
     * @param json objeto que se va a guardar en el fichero.
     * @param ruta ruta del directorio del fichero que se va a leer.
     * @param sobreescribir si es true se sobreescribirá el contenido fichero. Si es false
se respetará el contenido
     * y la información se añadirá al final del fichero.
     * @return código de retorno del constructor. Si finaliza con éxito devuelve 1.
     */
    public int saveData(String fichero, String json, final boolean sobreescribir, String ruta
) {
        final String writeFile, writeContents;
        try {
            writeFile = "file://"+ruta+fichero;
            writeContents = json;
            Thread t = new Thread(){
                public void run(){
                    try {
                        OutputStream os;
                        FileConnection fc = (FileConnection) Connector.open(writeFile,
Connector.WRITE);
                        if (!fc.exists())
                            fc.create();
                        if (sobreescribir) {
                            os = fc.openOutputStream();//empieza a escribir al principio del
fichero
                        }
                        else
                            os = fc.openOutputStream(fc.fileSize());//le decimos que empiece
al final
                        PrintStream ps = new PrintStream(os);
                        ps.println(writeContents);
                        fc.close();
                        exito = 1;
                    } catch (IOException e){
```

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

```
                    e.printStackTrace();
                    exito = -1;
                } catch (SecurityException e){
                    e.printStackTrace();
                    exito = -2;
                }
            }
        };
        t.start();
    } catch (Exception e) {
        e.printStackTrace();
        exito = -1;
    }
    return exito;
}

/**
 * Constructor de la clase saveData. Guarda un vector de json en forma de String en un
fichero de texto.
 * Tiene la opción de sobrescribir el fichero o escribir al final de éste.
 * @param fichero nombre del fichero donde se va a guardar la información.
 * @param ruta ruta del directorio del fichero que se va a leer.
 * @param sobreescribir si es true se sobreescribirá el contenido fichero. Si es false
se respetará el contenido
 * y la información se añadirá al final del fichero.
 * @param vjson vector cuyos elementos serán guardados en el fichero separados por
saltos de línea.
 * @return código de retorno del constructor. Si finaliza con éxito devuelve 1.
 */
public int saveData(String fichero, Vector vjson, final boolean sobreescribir, String
ruta) {
    final String writeFile;
    writeFile = "file://"+ruta+fichero;
    try {
        OutputStream os;
        FileConnection fc = (FileConnection) Connector.open(writeFile, Connector.WRITE);
        if (!fc.exists())
            fc.create();
        if (sobreescribir) {
            fc.truncate(0);//elimina el contenido del fichero registros
            os = fc.openOutputStream();
            vjson.trimToSize();
            fc.fileSize();
        }
        else
            os = fc.openOutputStream(fc.fileSize());//para que empiece a escribir al
final del fichero
        PrintStream ps = new PrintStream(os);
        for (int i=0; i<vjson.size(); i++)
            ps.println(vjson.elementAt(i).toString());
        fc.close();
        exito = 1;
    }
    catch (IOException e) {
        e.printStackTrace();
        exito = -1;
    }
```

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

```
        catch (SecurityException se) {
            se.printStackTrace();
            exito = -2;
        }
        return exito;
    }

    /**
     * Constructor de la clase saveData. Guarda un array de objetos JSON en un fichero de
texto.
     * Tiene la opción de sobrescribir el fichero o escribir al final de éste.
     * @param fichero nombre del fichero donde se va a guardar la información.
     * @param jObject array de objetos JSON cuyos elementos serán guardados en el fichero
separados por saltos de línea.
     * @param ruta ruta del directorio del fichero que se va a leer.
     * @param sobreescribir si es true se sobreescribirá el contenido fichero. Si es false
se respetará el contenido
     * y la información se añadirá al final del fichero.
     * @return código de retorno del constructor. Si finaliza con éxito devuelve 1.
     */
    public int saveData(String fichero, JSONObject[] jObject, final boolean sobreescribir,
String ruta) {
        final String writeFile;
        Vector v = new Vector();
        for (int i=0; i<jObject.length; i++) {
            v.addElement(jObject[i].toString());
        }
        writeFile = "file://"+ruta+fichero;
        try {
            OutputStream os;
            FileConnection fc = (FileConnection) Connector.open(writeFile, Connector.WRITE);
            if (!fc.exists())
                fc.create();
            if (sobreescribir)
                os = fc.openOutputStream();
            else
                os = fc.openOutputStream(fc.fileSize());//empieza a escribir al final del
fichero
            PrintStream ps = new PrintStream(os);
            for (int i=0; i<v.size(); i++)
                ps.println(v.elementAt(i).toString());
            fc.close();
            exito = 1;
        }
        catch (IOException e){
            e.printStackTrace();
            exito = -1;
        }
        return exito;
    }

    /**
     * Elimina el fichero
     * @param fichero nombre del fichero que se desea elminar
     */
    public void delete(String fichero) {
        try {
```

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

```
            String delFile = System.getProperty("fileconn.dir.memorycard")+fichero;
            FileConnection fc = (FileConnection) Connector.open(delFile, Connector.READ_WRITE
);
            fc.delete();
            fc.close();
        }
        catch (IOException e) {
            e.printStackTrace();
        }
    }
}
```

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

```
cod_operation
cod_base
cod_device
http://130.206.164.125/emergency/input.php
m=
records.txt
/e:/
es
```

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

Fever unkwnown origin
Acute respiratory infections
Malaria (suspected)
Malaria (confirm)
Typhoid fever
Haemorrhagic fever
Jaundice
Measles
Watery diarrhoea
Bloody diarrhoea
Tetanus
Wound/trauma
Anthrax
Cutaneous infection
Malnutrition
Vaginal infection
Others

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

# Anexo II: Javadoc



upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

**main**

# Class menuMIDlet

```
java.lang.Object
  └javax.microedition.midlet.MIDlet
      └main.menuMIDlet
```

**All Implemented Interfaces:**

javax.microedition.lcdui.CommandListener

---

```
public class menuMIDlet
extends javax.microedition.midlet.MIDlet
implements javax.microedition.lcdui.CommandListener
```

Contiene el MIDlet de la aplicación con la interfaz gráfica para acceder a las funcionalidades del programa.

**Version:**

1.0

**Author:**

Daniel Mayor

---

## Constructor Summary

| |
|---|
| **menuMIDlet**()<br>The menuMIDlet MIDlet constructor. |

## Method Summary

| | |
|---|---|
| void | **bucle_addAction**()<br>Performs an action assigned to the selected list element in the bucle_add component. |
| void | **bucle_editAction**()<br>Performs an action assigned to the selected list element in the bucle_edit component. |
| void | **commandAction**(javax.microedition.lcdui.Command command, javax.microedition.lcdui.Displayable displayable)<br>Called by a system to indicated that a command has been invoked on a particular displayable. |
| void | **destroyApp**(boolean unconditional)<br>Called to signal the MIDlet to terminate. |
| void | **exitMIDlet**()<br>Exits MIDlet. |
| javax.microedition.lcdui.ChoiceGroup | **generaChoiceGroup**(java.util.Vector vjson, javax.microedition.lcdui.ChoiceGroup choiceGroup)<br>Genera un ChoiceGroup con las fechas de los registros almacenados en el fichero de texto |
| java.lang.String | **generaDate**(java.util.Date fecha)<br>Genera un String en formato yyyyMMdd a partir del objeto fecha |
| java.lang.String | **generaJSON**()<br>Genera el String JSON |
| javax.microedition.lcdui.Command | **getBackCommand_add**()<br>Returns an initiliazed instance of backCommand_add component. |
| javax.microedition.lcdui.Command | **getBackCommand_config**()<br>Returns an initiliazed instance of backCommand_config component. |
| javax.microedition.lcdui.Command | **getBackCommand_del**()<br>Returns an initiliazed instance of backCommand_del component. |
| javax.microedition.lcdui.Command | **getBackCommand_edit_number**()<br>Returns an initiliazed instance of backCommand_edit_number component. |
| javax.microedition.lcdui.Command | **getBackCommand_edit_select_symptom**()<br>Returns an initiliazed instance of backCommand_edit_select_symptom component. |

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

| | |
|---|---|
| javax.microedition.lcdui.Command | **getBackCommand_edit**()<br>Returns an initiliazed instance of backCommand_edit component. |
| javax.microedition.lcdui.Command | **getBackCommand_number**()<br>Returns an initiliazed instance of backCommand_number component. |
| javax.microedition.lcdui.Command | **getBackCommand_select_symptom**()<br>Returns an initiliazed instance of backCommand_select_symptom component. |
| javax.microedition.lcdui.Command | **getBackCommand_textBox_del**()<br>Returns an initiliazed instance of backCommand_textBox_del component. |
| javax.microedition.lcdui.Command | **getBackCommand_textBox_edit**()<br>Returns an initiliazed instance of backCommand_textBox_edit component. |
| javax.microedition.lcdui.Command | **getBackCommand_textBox_view**()<br>Returns an initiliazed instance of backCommand_textBox_view component. |
| javax.microedition.lcdui.Command | **getBackCommand_transfer**()<br>Returns an initiliazed instance of backCommand_transfer component. |
| javax.microedition.lcdui.Command | **getBackCommand_view**()<br>Returns an initiliazed instance of backCommand_view component. |
| javax.microedition.lcdui.List | **getBucle_add**()<br>Returns an initiliazed instance of bucle_add component. |
| javax.microedition.lcdui.List | **getBucle_edit**()<br>Returns an initiliazed instance of bucle_edit component. |
| javax.microedition.lcdui.ChoiceGroup | **getChoiceGroup_del**()<br>Returns an initiliazed instance of choiceGroup_del component. |
| javax.microedition.lcdui.ChoiceGroup | **getChoiceGroup_edit_symptom**()<br>Returns an initiliazed instance of choiceGroup_edit_symptom component. |
| javax.microedition.lcdui.ChoiceGroup | **getChoiceGroup_edit**()<br>Returns an initiliazed instance of choiceGroup_edit component. |
| javax.microedition.lcdui.ChoiceGroup | **getChoiceGroup_symptom**()<br>Returns an initiliazed instance of choiceGroup_symptom component. |
| javax.microedition.lcdui.ChoiceGroup | **getChoiceGroup_view**()<br>Returns an initiliazed instance of choiceGroup_view component. |
| javax.microedition.lcdui.DateField | **getDateField_add**()<br>Returns an initiliazed instance of dateField_add component. |
| javax.microedition.lcdui.Display | **getDisplay**()<br>Returns a display instance. |
| javax.microedition.lcdui.Command | **getExitCommand_main**()<br>Returns an initiliazed instance of exitCommand_main component. |
| javax.microedition.lcdui.Form | **getForm_add_number**()<br>Returns an initiliazed instance of form_add_number component. |
| javax.microedition.lcdui.Form | **getForm_add_select_symptom**()<br>Returns an initiliazed instance of form_add_select_symptom component. |
| javax.microedition.lcdui.Form | **getForm_add**()<br>Returns an initiliazed instance of form_add component. |
| javax.microedition.lcdui.Form | **getForm_del**()<br>Returns an initiliazed instance of form_del component. |
| javax.microedition.lcdui.Form | **getForm_edit_number**()<br>Returns an initiliazed instance of form_edit_number component. |
| javax.microedition.lcdui.Form | **getForm_edit_select_symptom**()<br>Returns an initiliazed instance of form_edit_select_symptom component. |
| javax.microedition.lcdui.Form | **getForm_edit**()<br>Returns an initiliazed instance of form_edit component. |
| javax.microedition.lcdui.Form | **getForm_settings**()<br>Returns an initiliazed instance of form_settings component. |
| javax.microedition.lcdui.Form | **getForm_view**()<br>Returns an initiliazed instance of form_view component. |

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

| | |
|---|---|
| javax.microedition.lcdui.TextField | **getID_Base_field**()<br>Returns an initiliazed instance of ID_Base_field component. |
| javax.microedition.lcdui.TextField | **getID_Device_field**()<br>Returns an initiliazed instance of ID_Device_field component. |
| javax.microedition.lcdui.TextField | **getID_Op_field**()<br>Returns an initiliazed instance of ID_Op_field component. |
| javax.microedition.lcdui.List | **getMain**()<br>Returns an initiliazed instance of main component. |
| javax.microedition.lcdui.Command | **getOkCommand_add**()<br>Returns an initiliazed instance of okCommand_add component. |
| javax.microedition.lcdui.Command | **getOkCommand_bucle_add**()<br>Returns an initiliazed instance of okCommand_bucle_add component. |
| javax.microedition.lcdui.Command | **getOkCommand_bucle_edit**()<br>Returns an initiliazed instance of okCommand_bucle_edit component. |
| javax.microedition.lcdui.Command | **getOkCommand_config**()<br>Returns an initiliazed instance of okCommand_config component. |
| javax.microedition.lcdui.Command | **getOkCommand_del**()<br>Returns an initiliazed instance of okCommand_del component. |
| javax.microedition.lcdui.Command | **getOkCommand_edit_number**()<br>Returns an initiliazed instance of okCommand_edit_number component. |
| javax.microedition.lcdui.Command | **getOkCommand_edit_select_symptom**()<br>Returns an initiliazed instance of okCommand_edit_select_symptom component. |
| javax.microedition.lcdui.Command | **getOkCommand_edit**()<br>Returns an initiliazed instance of okCommand_edit component. |
| javax.microedition.lcdui.Command | **getOkCommand_form_set**()<br>Returns an initiliazed instance of okCommand_form_set component. |
| javax.microedition.lcdui.Command | **getOkCommand_main**()<br>Returns an initiliazed instance of okCommand_main component. |
| javax.microedition.lcdui.Command | **getOkCommand_number**()<br>Returns an initiliazed instance of okCommand_number component. |
| javax.microedition.lcdui.Command | **getOkCommand_select_symptom**()<br>Returns an initiliazed instance of okCommand_select_symptom component. |
| javax.microedition.lcdui.Command | **getOkCommand_textBox_add**()<br>Returns an initiliazed instance of okCommand_textBox_add component. |
| javax.microedition.lcdui.Command | **getOkCommand_textBox_del**()<br>Returns an initiliazed instance of okCommand_textBox_del component. |
| javax.microedition.lcdui.Command | **getOkCommand_textBox_edit**()<br>Returns an initiliazed instance of okCommand_textBox_edit component. |
| javax.microedition.lcdui.Command | **getOkCommand_textBox_view**()<br>Returns an initiliazed instance of okCommand_textBox_view component. |
| javax.microedition.lcdui.Command | **getOkCommand_transfer**()<br>Returns an initiliazed instance of okCommand_transfer component. |
| javax.microedition.lcdui.Command | **getOkCommand_upload_Base**()<br>Returns an initiliazed instance of okCommand_upload_Base component. |
| javax.microedition.lcdui.Command | **getOkCommand_upload_HQ**()<br>Returns an initiliazed instance of okCommand_upload_HQ component. |
| javax.microedition.lcdui.Command | **getOkCommand_upload_SD**()<br>Returns an initiliazed instance of okCommand_upload_SD component. |
| javax.microedition.lcdui.Command | **getOkCommand_view**()<br>Returns an initiliazed instance of okCommand_view component. |
| javax.microedition.lcdui.List | **getSettings**()<br>Returns an initiliazed instance of settings component. |

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

| | |
|---|---|
| org.netbeans.microedition.util.SimpleCancellableTask | **getTask_edit**()<br>Returns an initiliazed instance of task_edit component. |
| org.netbeans.microedition.util.SimpleCancellableTask | **getTask_upload_Base**()<br>Returns an initiliazed instance of task_upload_Base component. |
| org.netbeans.microedition.util.SimpleCancellableTask | **getTask_upload_HQ**()<br>Returns an initiliazed instance of task_upload_HQ component. |
| org.netbeans.microedition.util.SimpleCancellableTask | **getTask_upload_SD**()<br>Returns an initiliazed instance of task_upload_SD component. |
| javax.microedition.lcdui.TextBox | **getTextBox_add**()<br>Returns an initiliazed instance of textBox_add component. |
| javax.microedition.lcdui.TextBox | **getTextBox_del**()<br>Returns an initiliazed instance of textBox_del component. |
| javax.microedition.lcdui.TextBox | **getTextBox_edit**()<br>Returns an initiliazed instance of textBox_edit component. |
| javax.microedition.lcdui.TextBox | **getTextBox_upload_Base**()<br>Returns an initiliazed instance of textBox_upload_Base component. |
| javax.microedition.lcdui.TextBox | **getTextBox_upload_HQ**()<br>Returns an initiliazed instance of textBox_upload_HQ component. |
| javax.microedition.lcdui.TextBox | **getTextBox_upload_SD**()<br>Returns an initiliazed instance of textBox_upload_SD component. |
| javax.microedition.lcdui.TextBox | **getTextBox_view**()<br>Returns an initiliazed instance of textBox_view component. |
| javax.microedition.lcdui.TextField | **getTextField_ed_fo5**()<br>Returns an initiliazed instance of textField_ed_fo5 component. |
| javax.microedition.lcdui.TextField | **getTextField_ed_fu5**()<br>Returns an initiliazed instance of textField_ed_fu5 component. |
| javax.microedition.lcdui.TextField | **getTextField_ed_mo5**()<br>Returns an initiliazed instance of textField_ed_mo5 component. |
| javax.microedition.lcdui.TextField | **getTextField_ed_mu5**()<br>Returns an initiliazed instance of textField_ed_mu5 component. |
| javax.microedition.lcdui.TextField | **getTextField_fo5**()<br>Returns an initiliazed instance of textField_fo5 component. |
| javax.microedition.lcdui.TextField | **getTextField_fu5**()<br>Returns an initiliazed instance of textField_fu5 component. |
| javax.microedition.lcdui.TextField | **getTextField_mo5**()<br>Returns an initiliazed instance of textField_mo5 component. |
| javax.microedition.lcdui.TextField | **getTextField_mu5**()<br>Returns an initiliazed instance of textField_mu5 component. |
| javax.microedition.lcdui.List | **getTransfer**()<br>Returns an initiliazed instance of Transfer component. |
| org.netbeans.microedition.lcdui.WaitScreen | **getWaitScreen_Base**()<br>Returns an initiliazed instance of waitScreen_Base component. |
| org.netbeans.microedition.lcdui.WaitScreen | **getWaitScreen_HQ**()<br>Returns an initiliazed instance of waitScreen_HQ component. |
| org.netbeans.microedition.lcdui.WaitScreen | **getWaitScreen_SD**()<br>Returns an initiliazed instance of waitScreen_SD component. |
| void | **mainAction**()<br>Performs an action assigned to the selected list element in the main component. |
| void | **pauseApp**()<br>Called when MIDlet is paused. |
| void | **resetForm_add**()<br>Resetea los formularios form_add_select_symptom, form_add_number y sus componentes |
| void | **resetForm_del**()<br>Resetea el formulario form_del y sus componentes |
| void | **resetForm_edit**()<br>Resetea el formulario form_edit y sus componentes |
| void | **resumeMIDlet**()<br>Performs an action assigned to the Mobile Device - MIDlet Resumed point. |
| void | **settingsAction**()<br>Performs an action assigned to the selected list element in the settings component. |

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

| | |
|---|---|
| void | **startApp**()<br>Called when MIDlet is started. |
| void | **startMIDlet**()<br>Performs an action assigned to the Mobile Device - MIDlet Started point. |
| void | **switchDisplayable**(javax.microedition.lcdui.Alert alert, javax.microedition.lcdui.Displayable nextDisplayable)<br>Switches a current displayable in a display. |
| void | **TransferAction**()<br>Performs an action assigned to the selected list element in the Transfer component. |

| Methods inherited from class javax.microedition.midlet.MIDlet |
|---|
| checkPermission, getAppProperty, notifyDestroyed, notifyPaused, platformRequest, resumeRequest |

| Methods inherited from class java.lang.Object |
|---|
| equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait |

# Constructor Detail

### menuMIDlet

```
public menuMIDlet()
```

The menuMIDlet MIDlet constructor.

# Method Detail

### startMIDlet

```
public void startMIDlet()
```

Performs an action assigned to the Mobile Device - MIDlet Started point.

---

### resumeMIDlet

```
public void resumeMIDlet()
```

Performs an action assigned to the Mobile Device - MIDlet Resumed point.

---

### switchDisplayable

```
public void switchDisplayable(javax.microedition.lcdui.Alert alert,
                              javax.microedition.lcdui.Displayable nextDisplayable)
```

Switches a current displayable in a display. The `display` instance is taken from `getDisplay` method. This method is used by all actions in the design for switching displayable.

**Parameters:**

`alert` - the Alert which is temporarily set to the display; if `null`, then `nextDisplayable` is set immediately
`nextDisplayable` - the Displayable to be set

---

### commandAction

```
public void commandAction(javax.microedition.lcdui.Command command,
                          javax.microedition.lcdui.Displayable displayable)
```

Called by a system to indicated that a command has been invoked on a particular displayable.

**Specified by:**

`commandAction` in interface `javax.microedition.lcdui.CommandListener`

**Parameters:**

`command` - the Command that was invoked
`displayable` - the Displayable where the command was invoked

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

### getMain

```
public javax.microedition.lcdui.List getMain()
```

Returns an initiliazed instance of main component.

**Returns:**
the initialized component instance

---

### mainAction

```
public void mainAction()
```

Performs an action assigned to the selected list element in the main component.

---

### getExitCommand_main

```
public javax.microedition.lcdui.Command getExitCommand_main()
```

Returns an initiliazed instance of exitCommand_main component.

**Returns:**
the initialized component instance

---

### getForm_add

```
public javax.microedition.lcdui.Form getForm_add()
```

Returns an initiliazed instance of form_add component.

**Returns:**
the initialized component instance

---

### getDateField_add

```
public javax.microedition.lcdui.DateField getDateField_add()
```

Returns an initiliazed instance of dateField_add component.

**Returns:**
the initialized component instance

---

### getForm_view

```
public javax.microedition.lcdui.Form getForm_view()
```

Returns an initiliazed instance of form_view component.

**Returns:**
the initialized component instance

---

### getForm_edit

```
public javax.microedition.lcdui.Form getForm_edit()
```

Returns an initiliazed instance of form_edit component.

**Returns:**
the initialized component instance

---

### getOkCommand_main

```
public javax.microedition.lcdui.Command getOkCommand_main()
```

Returns an initiliazed instance of okCommand_main component.

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

**Returns:**
the initialized component instance

---

### getOkCommand_add

```
public javax.microedition.lcdui.Command getOkCommand_add()
```

Returns an initiliazed instance of okCommand_add component.

**Returns:**
the initialized component instance

---

### getBackCommand_view

```
public javax.microedition.lcdui.Command getBackCommand_view()
```

Returns an initiliazed instance of backCommand_view component.

**Returns:**
the initialized component instance

---

### getBackCommand_add

```
public javax.microedition.lcdui.Command getBackCommand_add()
```

Returns an initiliazed instance of backCommand_add component.

**Returns:**
the initialized component instance

---

### getOkCommand_edit

```
public javax.microedition.lcdui.Command getOkCommand_edit()
```

Returns an initiliazed instance of okCommand_edit component.

**Returns:**
the initialized component instance

---

### getBackCommand_edit

```
public javax.microedition.lcdui.Command getBackCommand_edit()
```

Returns an initiliazed instance of backCommand_edit component.

**Returns:**
the initialized component instance

---

### getTransfer

```
public javax.microedition.lcdui.List getTransfer()
```

Returns an initiliazed instance of Transfer component.

**Returns:**
the initialized component instance

---

### TransferAction

```
public void TransferAction()
```

Performs an action assigned to the selected list element in the Transfer component.

---

### getBackCommand_transfer

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

```
public javax.microedition.lcdui.Command getBackCommand_transfer()
```

Returns an initiliazed instance of backCommand_transfer component.

**Returns:**
the initialized component instance

---

### getSettings

```
public javax.microedition.lcdui.List getSettings()
```

Returns an initiliazed instance of settings component.

**Returns:**
the initialized component instance

---

### settingsAction

```
public void settingsAction()
```

Performs an action assigned to the selected list element in the settings component.

---

### getBackCommand_config

```
public javax.microedition.lcdui.Command getBackCommand_config()
```

Returns an initiliazed instance of backCommand_config component.

**Returns:**
the initialized component instance

---

### getChoiceGroup_symptom

```
public javax.microedition.lcdui.ChoiceGroup getChoiceGroup_symptom()
```

Returns an initiliazed instance of choiceGroup_symptom component.

**Returns:**
the initialized component instance

---

### getBackCommand_select_symptom

```
public javax.microedition.lcdui.Command getBackCommand_select_symptom()
```

Returns an initiliazed instance of backCommand_select_symptom component.

**Returns:**
the initialized component instance

---

### getForm_add_number

```
public javax.microedition.lcdui.Form getForm_add_number()
```

Returns an initiliazed instance of form_add_number component.

**Returns:**
the initialized component instance

---

### getTextField_mu5

```
public javax.microedition.lcdui.TextField getTextField_mu5()
```

Returns an initiliazed instance of textField_mu5 component.

**Returns:**
the initialized component instance

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

### getOkCommand_number

```
public javax.microedition.lcdui.Command getOkCommand_number()
```

Returns an initiliazed instance of okCommand_number component.

**Returns:**
the initialized component instance

---

### getOkCommand_select_symptom

```
public javax.microedition.lcdui.Command getOkCommand_select_symptom()
```

Returns an initiliazed instance of okCommand_select_symptom component.

**Returns:**
the initialized component instance

---

### getTextField_mo5

```
public javax.microedition.lcdui.TextField getTextField_mo5()
```

Returns an initiliazed instance of textField_mo5 component.

**Returns:**
the initialized component instance

---

### getTextField_fu5

```
public javax.microedition.lcdui.TextField getTextField_fu5()
```

Returns an initiliazed instance of textField_fu5 component.

**Returns:**
the initialized component instance

---

### getTextField_fo5

```
public javax.microedition.lcdui.TextField getTextField_fo5()
```

Returns an initiliazed instance of textField_fo5 component.

**Returns:**
the initialized component instance

---

### getBackCommand_number

```
public javax.microedition.lcdui.Command getBackCommand_number()
```

Returns an initiliazed instance of backCommand_number component.

**Returns:**
the initialized component instance

---

### getTextBox_add

```
public javax.microedition.lcdui.TextBox getTextBox_add()
```

Returns an initiliazed instance of textBox_add component.

**Returns:**
the initialized component instance

---

### getOkCommand_textBox_add

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

```
public javax.microedition.lcdui.Command getOkCommand_textBox_add()
```

Returns an initiliazed instance of okCommand_textBox_add component.

**Returns:**
the initialized component instance

---

### getForm_add_select_symptom

```
public javax.microedition.lcdui.Form getForm_add_select_symptom()
```

Returns an initiliazed instance of form_add_select_symptom component.

**Returns:**
the initialized component instance

---

### getOkCommand_bucle_add

```
public javax.microedition.lcdui.Command getOkCommand_bucle_add()
```

Returns an initiliazed instance of okCommand_bucle_add component.

**Returns:**
the initialized component instance

---

### getBucle_add

```
public javax.microedition.lcdui.List getBucle_add()
```

Returns an initiliazed instance of bucle_add component.

**Returns:**
the initialized component instance

---

### bucle_addAction

```
public void bucle_addAction()
```

Performs an action assigned to the selected list element in the bucle_add component.

---

### getTextBox_view

```
public javax.microedition.lcdui.TextBox getTextBox_view()
```

Returns an initiliazed instance of textBox_view component.

**Returns:**
the initialized component instance

---

### getOkCommand_view

```
public javax.microedition.lcdui.Command getOkCommand_view()
```

Returns an initiliazed instance of okCommand_view component.

**Returns:**
the initialized component instance

---

### getOkCommand_textBox_view

```
public javax.microedition.lcdui.Command getOkCommand_textBox_view()
```

Returns an initiliazed instance of okCommand_textBox_view component.

**Returns:**
the initialized component instance

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

### getBackCommand_textBox_view

```
public javax.microedition.lcdui.Command getBackCommand_textBox_view()
```

Returns an initiliazed instance of backCommand_textBox_view component.

**Returns:**
the initialized component instance

---

### getOkCommand_config

```
public javax.microedition.lcdui.Command getOkCommand_config()
```

Returns an initiliazed instance of okCommand_config component.

**Returns:**
the initialized component instance

---

### getForm_del

```
public javax.microedition.lcdui.Form getForm_del()
```

Returns an initiliazed instance of form_del component.

**Returns:**
the initialized component instance

---

### getBackCommand_del

```
public javax.microedition.lcdui.Command getBackCommand_del()
```

Returns an initiliazed instance of backCommand_del component.

**Returns:**
the initialized component instance

---

### getOkCommand_del

```
public javax.microedition.lcdui.Command getOkCommand_del()
```

Returns an initiliazed instance of okCommand_del component.

**Returns:**
the initialized component instance

---

### getChoiceGroup_del

```
public javax.microedition.lcdui.ChoiceGroup getChoiceGroup_del()
```

Returns an initiliazed instance of choiceGroup_del component.

**Returns:**
the initialized component instance

---

### getForm_settings

```
public javax.microedition.lcdui.Form getForm_settings()
```

Returns an initiliazed instance of form_settings component.

**Returns:**
the initialized component instance

---

### getID_Op_field

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

```
public javax.microedition.lcdui.TextField getID_Op_field()
```

Returns an initiliazed instance of ID_Op_field component.

**Returns:**
the initialized component instance

---

### getID_Base_field

```
public javax.microedition.lcdui.TextField getID_Base_field()
```

Returns an initiliazed instance of ID_Base_field component.

**Returns:**
the initialized component instance

---

### getID_Device_field

```
public javax.microedition.lcdui.TextField getID_Device_field()
```

Returns an initiliazed instance of ID_Device_field component.

**Returns:**
the initialized component instance

---

### getOkCommand_form_set

```
public javax.microedition.lcdui.Command getOkCommand_form_set()
```

Returns an initiliazed instance of okCommand_form_set component.

**Returns:**
the initialized component instance

---

### getTextBox_del

```
public javax.microedition.lcdui.TextBox getTextBox_del()
```

Returns an initiliazed instance of textBox_del component.

**Returns:**
the initialized component instance

---

### getOkCommand_textBox_del

```
public javax.microedition.lcdui.Command getOkCommand_textBox_del()
```

Returns an initiliazed instance of okCommand_textBox_del component.

**Returns:**
the initialized component instance

---

### getBackCommand_textBox_del

```
public javax.microedition.lcdui.Command getBackCommand_textBox_del()
```

Returns an initiliazed instance of backCommand_textBox_del component.

**Returns:**
the initialized component instance

---

### getChoiceGroup_view

```
public javax.microedition.lcdui.ChoiceGroup getChoiceGroup_view()
```

Returns an initiliazed instance of choiceGroup_view component.

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

**Returns:**
the initialized component instance

---

### getTextBox_edit

```
public javax.microedition.lcdui.TextBox getTextBox_edit()
```

Returns an initiliazed instance of textBox_edit component.

**Returns:**
the initialized component instance

---

### getOkCommand_textBox_edit

```
public javax.microedition.lcdui.Command getOkCommand_textBox_edit()
```

Returns an initiliazed instance of okCommand_textBox_edit component.

**Returns:**
the initialized component instance

---

### getBackCommand_textBox_edit

```
public javax.microedition.lcdui.Command getBackCommand_textBox_edit()
```

Returns an initiliazed instance of backCommand_textBox_edit component.

**Returns:**
the initialized component instance

---

### getChoiceGroup_edit

```
public javax.microedition.lcdui.ChoiceGroup getChoiceGroup_edit()
```

Returns an initiliazed instance of choiceGroup_edit component.

**Returns:**
the initialized component instance

---

### getForm_edit_select_symptom

```
public javax.microedition.lcdui.Form getForm_edit_select_symptom()
```

Returns an initiliazed instance of form_edit_select_symptom component.

**Returns:**
the initialized component instance

---

### getOkCommand_edit_select_symptom

```
public javax.microedition.lcdui.Command getOkCommand_edit_select_symptom()
```

Returns an initiliazed instance of okCommand_edit_select_symptom component.

**Returns:**
the initialized component instance

---

### getBackCommand_edit_select_symptom

```
public javax.microedition.lcdui.Command getBackCommand_edit_select_symptom()
```

Returns an initiliazed instance of backCommand_edit_select_symptom component.

**Returns:**
the initialized component instance

---

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

### getChoiceGroup_edit_symptom

```
public javax.microedition.lcdui.ChoiceGroup getChoiceGroup_edit_symptom()
```

Returns an initiliazed instance of choiceGroup_edit_symptom component.

**Returns:**
the initialized component instance

---

### getForm_edit_number

```
public javax.microedition.lcdui.Form getForm_edit_number()
```

Returns an initiliazed instance of form_edit_number component.

**Returns:**
the initialized component instance

---

### getTextField_ed_mu5

```
public javax.microedition.lcdui.TextField getTextField_ed_mu5()
```

Returns an initiliazed instance of textField_ed_mu5 component.

**Returns:**
the initialized component instance

---

### getTextField_ed_mo5

```
public javax.microedition.lcdui.TextField getTextField_ed_mo5()
```

Returns an initiliazed instance of textField_ed_mo5 component.

**Returns:**
the initialized component instance

---

### getTextField_ed_fu5

```
public javax.microedition.lcdui.TextField getTextField_ed_fu5()
```

Returns an initiliazed instance of textField_ed_fu5 component.

**Returns:**
the initialized component instance

---

### getTextField_ed_fo5

```
public javax.microedition.lcdui.TextField getTextField_ed_fo5()
```

Returns an initiliazed instance of textField_ed_fo5 component.

**Returns:**
the initialized component instance

---

### getBucle_edit

```
public javax.microedition.lcdui.List getBucle_edit()
```

Returns an initiliazed instance of bucle_edit component.

**Returns:**
the initialized component instance

---

### bucle_editAction

```
public void bucle_editAction()
```

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

Performs an action assigned to the selected list element in the bucle_edit component.

---

### getOkCommand_edit_number

```
public javax.microedition.lcdui.Command getOkCommand_edit_number()
```

Returns an initiliazed instance of okCommand_edit_number component.

**Returns:**
the initialized component instance

---

### getBackCommand_edit_number

```
public javax.microedition.lcdui.Command getBackCommand_edit_number()
```

Returns an initiliazed instance of backCommand_edit_number component.

**Returns:**
the initialized component instance

---

### getOkCommand_bucle_edit

```
public javax.microedition.lcdui.Command getOkCommand_bucle_edit()
```

Returns an initiliazed instance of okCommand_bucle_edit component.

**Returns:**
the initialized component instance

---

### getTextBox_upload_HQ

```
public javax.microedition.lcdui.TextBox getTextBox_upload_HQ()
```

Returns an initiliazed instance of textBox_upload_HQ component.

**Returns:**
the initialized component instance

---

### getOkCommand_upload_HQ

```
public javax.microedition.lcdui.Command getOkCommand_upload_HQ()
```

Returns an initiliazed instance of okCommand_upload_HQ component.

**Returns:**
the initialized component instance

---

### getOkCommand_transfer

```
public javax.microedition.lcdui.Command getOkCommand_transfer()
```

Returns an initiliazed instance of okCommand_transfer component.

**Returns:**
the initialized component instance

---

### getWaitScreen_HQ

```
public org.netbeans.microedition.lcdui.WaitScreen getWaitScreen_HQ()
```

Returns an initiliazed instance of waitScreen_HQ component.

**Returns:**
the initialized component instance

---

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

### getTask_upload_HQ

```
public org.netbeans.microedition.util.SimpleCancellableTask getTask_upload_HQ()
```

Returns an initiliazed instance of task_upload_HQ component.

**Returns:**
the initialized component instance

---

### getTextBox_upload_Base

```
public javax.microedition.lcdui.TextBox getTextBox_upload_Base()
```

Returns an initiliazed instance of textBox_upload_Base component.

**Returns:**
the initialized component instance

---

### getTextBox_upload_SD

```
public javax.microedition.lcdui.TextBox getTextBox_upload_SD()
```

Returns an initiliazed instance of textBox_upload_SD component.

**Returns:**
the initialized component instance

---

### getWaitScreen_Base

```
public org.netbeans.microedition.lcdui.WaitScreen getWaitScreen_Base()
```

Returns an initiliazed instance of waitScreen_Base component.

**Returns:**
the initialized component instance

---

### getWaitScreen_SD

```
public org.netbeans.microedition.lcdui.WaitScreen getWaitScreen_SD()
```

Returns an initiliazed instance of waitScreen_SD component.

**Returns:**
the initialized component instance

---

### getOkCommand_upload_Base

```
public javax.microedition.lcdui.Command getOkCommand_upload_Base()
```

Returns an initiliazed instance of okCommand_upload_Base component.

**Returns:**
the initialized component instance

---

### getOkCommand_upload_SD

```
public javax.microedition.lcdui.Command getOkCommand_upload_SD()
```

Returns an initiliazed instance of okCommand_upload_SD component.

**Returns:**
the initialized component instance

---

### getTask_upload_Base

```
public org.netbeans.microedition.util.SimpleCancellableTask getTask_upload_Base()
```

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

Returns an initiliazed instance of task_upload_Base component.

**Returns:**
the initialized component instance

---

### getTask_upload_SD

```
public org.netbeans.microedition.util.SimpleCancellableTask getTask_upload_SD()
```

Returns an initiliazed instance of task_upload_SD component.

**Returns:**
the initialized component instance

---

### getTask_edit

```
public org.netbeans.microedition.util.SimpleCancellableTask getTask_edit()
```

Returns an initiliazed instance of task_edit component.

**Returns:**
the initialized component instance

---

### resetForm_add

```
public void resetForm_add()
```

Resetea los formularios form_add_select_symptom, form_add_number y sus componentes

---

### resetForm_del

```
public void resetForm_del()
```

Resetea el formulario form_del y sus componentes

---

### resetForm_edit

```
public void resetForm_edit()
```

Resetea el formulario form_edit y sus componentes

---

### generaChoiceGroup

```
public javax.microedition.lcdui.ChoiceGroup generaChoiceGroup(java.util.Vector vjson,
                                                              javax.microedition.lcdui.ChoiceGroup choiceGroup)
```

Genera un ChoiceGroup con las fechas de los registros almacenados en el fichero de texto

**Parameters:**
`vjson` - es el vector con el que se genera el ChoiceGroup
`choiceGroup` - es el nombre del ChoiceGroup sin elementos
**Returns:**
devuelve el ChoiceGroup con los elementos

---

### generaJSON

```
public java.lang.String generaJSON()
```

Genera el String JSON

**Returns:**
devuelve el JSON en un objeto String

---

### generaDate

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

```
public java.lang.String generaDate(java.util.Date fecha)
```

Genera un String en formato yyyyMMdd a partir del objeto fecha

**Parameters:**
`fecha` - objeto con el que se genera la fecha en String
**Returns:**
devuelve el String en formato yyyyMMdd

---

## getDisplay

```
public javax.microedition.lcdui.Display getDisplay()
```

Returns a display instance.

**Returns:**
the display instance.

---

## exitMIDlet

```
public void exitMIDlet()
```

Exits MIDlet.

---

## startApp

```
public void startApp()
```

Called when MIDlet is started. Checks whether the MIDlet have been already started and initialize/starts or resumes the MIDlet.

**Specified by:**
`startApp` in class `javax.microedition.midlet.MIDlet`

---

## pauseApp

```
public void pauseApp()
```

Called when MIDlet is paused.

**Specified by:**
`pauseApp` in class `javax.microedition.midlet.MIDlet`

---

## destroyApp

```
public void destroyApp(boolean unconditional)
```

Called to signal the MIDlet to terminate.

**Specified by:**
`destroyApp` in class `javax.microedition.midlet.MIDlet`
**Parameters:**
`unconditional` - if true, then the MIDlet has to be unconditionally terminated and all resources has to be released.

---



upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

**main**

# Class conectaHTTP

```
java.lang.Object
  └main.conectaHTTP
```

```
public class conectaHTTP
extends java.lang.Object
```

Realiza las conexiones con el clúster central mediante el protocolo HTTP.

**Author:**
Daniel

## Constructor Summary

| |
|---|
| conectaHTTP(java.lang.String stringjson, java.lang.String urlserver, java.lang.String urlquery)<br>Constructor de la clase conectaHTTP. |

## Method Summary

| | |
|---|---|
| java.lang.String | retorno()<br>Muestra la respuesta del servidor. |

| Methods inherited from class java.lang.Object |
|---|
| equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait |

## Constructor Detail

### conectaHTTP

```
public conectaHTTP(java.lang.String stringjson,
                   java.lang.String urlserver,
                   java.lang.String urlquery)
```

Constructor de la clase conectaHTTP.

**Parameters:**
`stringjson` - texto que se transmite vía POST al servidor.
`urlquery` - consulta que se envía al servidor en formato =.
`urlserver` - URL del servidor.

## Method Detail

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

## retorno

```
public java.lang.String retorno()
```

Muestra la respuesta del servidor.

**Returns:**
mensaje de retorno del servidor.

---




upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

**fileaccess**

# Class getConfigIDs

```
java.lang.Object
  └ fileaccess.getConfigIDs
```

```
public class getConfigIDs
extends java.lang.Object
```

Accede al fichero de configuración settings.txt y obtiene las variables globales con los ajustes del programa.

**Author:**
Daniel Mayor

## Constructor Summary

| | |
|---|---|
| **getConfigIDs**(java.lang.String fichero) | Constructor de la clase getConfigIDs. |

## Method Summary

| | |
|---|---|
| java.lang.String[] | **idstring**() Devuelve un array de Strings con el contenido del vector ids |
| JSONObject | **vector2JSON**() **Deprecated.** *no se utiliza* |

**Methods inherited from class java.lang.Object**

equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

## Constructor Detail

### getConfigIDs

```
public getConfigIDs(java.lang.String fichero)
```

Constructor de la clase getConfigIDs. Lee el fichero de configuración y almacena las variables en un vector.

**Parameters:**
`fichero` - nombre del fichero.

## Method Detail

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

## idstring

```
public java.lang.String[] idstring()
```

Devuelve un array de Strings con el contenido del vector ids

**Returns:**
idstring contiene los elementos del vector

---

## vector2JSON

```
public JSONObject vector2JSON()
```

**Deprecated.** *no se utiliza*

Convierte el vector con los json en un objeto JSON

**Returns:**
jObject objeto JSON

---



upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

**fileaccess**

# Class getConfigSymptom

```
java.lang.Object
  └ fileaccess.getConfigSymptom
```

```
public class getConfigSymptom
extends java.lang.Object
```

Lee del fichero de configuración symptoms.txt los síntomas de enfermedades que se analizarán.

**Author:**
Daniel Mayor

## Constructor Summary

| | |
|---|---|
| **getConfigSymptom**(java.lang.String fichero)<br>Constructor de la clase getConfigSymptom. | |

## Method Summary

| | |
|---|---|
| int | **n_sintomas**()<br>Devuelve el número de síntomas. |
| java.lang.String[] | **sintomas**()<br>Devuelve un array de Strings que almacena los síntomas. |

| Methods inherited from class java.lang.Object |
|---|
| equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait |

## Constructor Detail

### getConfigSymptom

```
public getConfigSymptom(java.lang.String fichero)
```

Constructor de la clase getConfigSymptom. Lee el fichero con los síntomas y los almacena en un vector.

**Parameters:**
`fichero` - nombre del fichero con los síntomas.

## Method Detail

### n_sintomas

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

`public int` **`n_sintomas`**`()`

Devuelve el número de síntomas.

**Returns:**
num_sintomas número de síntomas.

---

## sintomas

`public java.lang.String[]` **`sintomas`**`()`

Devuelve un array de Strings que almacena los síntomas.

**Returns:**
sintomas array de Strings con los síntomas.

---




upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa

Todos los derechos reservados
Eskubide guztiak erresalbatu dira

**fileaccess**

# Class getJSON

```
java.lang.Object
  └ fileaccess.getJSON
```

```
public class getJSON
extends java.lang.Object
```

Lee los ficheros de texto almacenados en el dispositivo móvil con los datos recogidos por el sistema.

**Author:**
Daniel Mayor

## Constructor Summary

| |
|---|
| **getJSON**(java.lang.String fichero)<br>Constructor de la clase getJSON. |
| **getJSON**(java.lang.String fichero, java.lang.String ruta)<br>Constructor de la clase getJSON. |

## Method Summary

| | |
|---|---|
| java.lang.String | **all**()<br>Devuelve un los elementos del vector concatenados y separados por un salto de línea |
| java.util.Vector | **getVjson**()<br>Devuelve el vector con los registros de los json almacenados. |
| java.lang.String[] | **jsons**()<br>Convierte el vector v en un array de Strings. |
| int | **n_jsons**()<br>Devuelve el tamaño del vector. |
| void | **ordenamientoBurbuja**(java.lang.String[] v, int util_v)<br>Algoritmo de ordenación por el método de la burbuja. |
| java.util.Vector | **sortVector**(java.util.Vector v, java.lang.String key)<br>Ordena los registros de los json almacenados en el vector de menor a mayor en función del key introducido con el algoritmo de la burbuja. |

| **Methods inherited from class java.lang.Object** |
|---|
| equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait |

## Constructor Detail

### getJSON

Todos los derechos reservados
Eskubide guztiak erresalbatu dira

```
public getJSON(java.lang.String fichero)
```

Constructor de la clase getJSON. Abre un fichero con objetos JSON, los ordena por fecha y los mete en un vector.

**Parameters:**

`fichero` - nombre del fichero.

---

### getJSON

```
public getJSON(java.lang.String fichero,
               java.lang.String ruta)
```

Constructor de la clase getJSON. Abre un fichero con objetos JSON, los ordena por fecha y los mete en un vector.

**Parameters:**

`fichero` - nombre del fichero.
`ruta` - ruta del directorio del fichero que se va a leer.

## Method Detail

### getVjson

```
public java.util.Vector getVjson()
```

Devuelve el vector con los registros de los json almacenados.

**Returns:**

v vector con los registros de los json almacenados.

---

### jsons

```
public java.lang.String[] jsons()
```

Convierte el vector v en un array de Strings.

**Returns:**

jsons array de Strings en el que almacena los json almacenados en el vector.

---

### n_jsons

```
public int n_jsons()
```

Devuelve el tamaño del vector.

**Returns:**

v.size() tamaño del vector.

---

### all

```
public java.lang.String all()
```

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

Devuelve un los elementos del vector concatenados y separados por un salto de línea

**Returns:**

cont String con los elementos del vector

---

## sortVector

```
public java.util.Vector sortVector(java.util.Vector v,
                                   java.lang.String key)
```

Ordena los registros de los json almacenados en el vector de menor a mayor en función del key introducido con el algoritmo de la burbuja.

**Parameters:**

`key` - nombre de la variable que se utiliza para realizar la ordenación.

`v` - vector que se va a ordenar.

**Returns:**

v_sorted vector ordenado.

---

## ordenamientoBurbuja

```
public void ordenamientoBurbuja(java.lang.String[] v,
                                int util_v)
```

Algoritmo de ordenación por el método de la burbuja.

**Parameters:**

`util_v` - número de elementos del conjunto que se va a ordenar.

`v` - elementos que se van a ordenar.

---




upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa

Todos los derechos reservados

Eskubide guztiak erresalbatu dira

**fileaccess**

# Class getJSON_jar

```
java.lang.Object
  └ fileaccess.getJSON_jar
```

```
public class getJSON_jar
extends java.lang.Object
```

Lee los ficheros de texto almacenados en el archivo .jar del dispositivo móvil con los datos recogidos por el sistema.

**Author:**
Daniel Mayor

## Constructor Summary

| |
|---|
| **getJSON_jar**(java.lang.String fichero)<br>Constructor de la clase getJSON_jar. |

## Method Summary

| | |
|---|---|
| java.lang.String[] | **jsons**()<br>Devuelve el vector en forma de array de Strings. |
| int | **n_jsons**()<br>Devuelve el tamaño del vector. |

**Methods inherited from class java.lang.Object**

equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

## Constructor Detail

### getJSON_jar

```
public getJSON_jar(java.lang.String fichero)
```

Constructor de la clase getJSON_jar. Lee los ficheros de texto almacenados dentro del .jar.

**Parameters:**
`fichero` - nombre del fichero.

## Method Detail

### jsons

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

`public java.lang.String[] jsons()`

Devuelve el vector en forma de array de Strings.

**Returns:**
jsons array de Strings con los JSON almacenados en el vector.

---

## n_jsons

`public int n_jsons()`

Devuelve el tamaño del vector.

**Returns:**
v.size() número de json almacenados en el fichero

---




upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa

Todos los derechos reservados
Eskubide guztiak erresalbatu dira

**fileaccess**

# Class saveData

```
java.lang.Object
  └ fileaccess.saveData
```

```
public class saveData
extends java.lang.Object
```

Contiene métodos para guardar los datos recogidos con el programa en ficheros de texto almacenados en el dispositivo móvil.

**Author:**
Daniel Mayor

## Constructor Summary

| |
|---|
| **saveData**() |

## Method Summary

| | |
|---|---|
| void | **delete**(java.lang.String fichero)<br>Elimina el fichero |
| int | **saveData**(java.lang.String fichero, JSONObject[] jObject, boolean sobreescribir)<br>Constructor de la clase saveData. |
| int | **saveData**(java.lang.String fichero, JSONObject[] jObject, boolean sobreescribir, java.lang.String ruta)<br>Constructor de la clase saveData. |
| int | **saveData**(java.lang.String fichero, java.lang.String json, boolean sobreescribir)<br>Constructor de la clase saveData. |
| int | **saveData**(java.lang.String fichero, java.lang.String json, boolean sobreescribir, java.lang.String ruta)<br>Constructor de la clase saveData. |
| int | **saveData**(java.lang.String fichero, java.util.Vector vjson, boolean sobreescribir)<br>Constructor de la clase saveData. |
| int | **saveData**(java.lang.String fichero, java.util.Vector vjson, boolean sobreescribir, java.lang.String ruta)<br>Constructor de la clase saveData. |

### Methods inherited from class java.lang.Object

equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

# Constructor Detail

### saveData

```
public saveData()
```

# Method Detail

### saveData

```
public int saveData(java.lang.String fichero,
                    java.lang.String json,
                    boolean sobreescribir)
```

Constructor de la clase saveData. Guarda un json en forma de String en un fichero de texto. Tiene la opción de sobrescribir el fichero o escribir al final de éste.

**Parameters:**

`fichero` - nombre del fichero donde se va a guardar la información.
`json` - objeto que se va a guardar en el fichero.
`sobreescribir` - si es true se sobreescribirá el contenido fichero. Si es false se respetará el contenido y la información se añadirá al final del fichero.

**Returns:**

código de retorno del constructor. Si finaliza con éxito devuelve 1.

---

### saveData

```
public int saveData(java.lang.String fichero,
                    java.util.Vector vjson,
                    boolean sobreescribir)
```

Constructor de la clase saveData. Guarda un vector de json en forma de String en un fichero de texto. Tiene la opción de sobrescribir el fichero o escribir al final de éste.

**Parameters:**

`fichero` - nombre del fichero donde se va a guardar la información.
`sobreescribir` - si es true se sobreescribirá el contenido fichero. Si es false se respetará el contenido y la información se añadirá al final del fichero.
`vjson` - vector cuyos elementos serán guardados en el fichero separados por saltos de línea.

**Returns:**

código de retorno del constructor. Si finaliza con éxito devuelve 1.

---

### saveData

```
public int saveData(java.lang.String fichero,
                    JSONObject[] jObject,
                    boolean sobreescribir)
```

Constructor de la clase saveData. Guarda un array de objetos JSON en un fichero de texto. Tiene la opción de sobrescribir el fichero o escribir al final de éste.

**Parameters:**

`fichero` - nombre del fichero donde se va a guardar la información.

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

`jObject` - array de objetos JSON cuyos elementos serán guardados en el fichero separados por saltos de línea.
`sobreescribir` - si es true se sobreescribirá el contenido fichero. Si es false se respetará el contenido y la información se añadirá al final del fichero.

**Returns:**

código de retorno del constructor. Si finaliza con éxito devuelve 1.

---

## saveData

```
public int saveData(java.lang.String fichero,
                    java.lang.String json,
                    boolean sobreescribir,
                    java.lang.String ruta)
```

Constructor de la clase saveData. Guarda un json en forma de String en un fichero de texto. Tiene la opción de sobrescribir el fichero o escribir al final de éste.

**Parameters:**

`fichero` - nombre del fichero donde se va a guardar la información.
`json` - objeto que se va a guardar en el fichero.
`ruta` - ruta del directorio del fichero que se va a leer.
`sobreescribir` - si es true se sobreescribirá el contenido fichero. Si es false se respetará el contenido y la información se añadirá al final del fichero.

**Returns:**

código de retorno del constructor. Si finaliza con éxito devuelve 1.

---

## saveData

```
public int saveData(java.lang.String fichero,
                    java.util.Vector vjson,
                    boolean sobreescribir,
                    java.lang.String ruta)
```

Constructor de la clase saveData. Guarda un vector de json en forma de String en un fichero de texto. Tiene la opción de sobrescribir el fichero o escribir al final de éste.

**Parameters:**

`fichero` - nombre del fichero donde se va a guardar la información.
`ruta` - ruta del directorio del fichero que se va a leer.
`sobreescribir` - si es true se sobreescribirá el contenido fichero. Si es false se respetará el contenido y la información se añadirá al final del fichero.
`vjson` - vector cuyos elementos serán guardados en el fichero separados por saltos de línea.

**Returns:**

código de retorno del constructor. Si finaliza con éxito devuelve 1.

---

## saveData

```
public int saveData(java.lang.String fichero,
                    JSONObject[] jObject,
                    boolean sobreescribir,
                    java.lang.String ruta)
```

Constructor de la clase saveData. Guarda un array de objetos JSON en un fichero de texto. Tiene la opción de sobrescribir el fichero o escribir al final de éste.

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

**Parameters:**

`fichero` - nombre del fichero donde se va a guardar la información.
`jObject` - array de objetos JSON cuyos elementos serán guardados en el fichero separados por saltos de línea.
`ruta` - ruta del directorio del fichero que se va a leer.
`sobreescribir` - si es true se sobreescribirá el contenido fichero. Si es false se respetará el contenido y la información se añadirá al final del fichero.

**Returns:**

código de retorno del constructor. Si finaliza con éxito devuelve 1.

---

## delete

```
public void delete(java.lang.String fichero)
```

Elimina el fichero

**Parameters:**

`fichero` - nombre del fichero que se desea elminar

---



upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

# Anexo III: Manual de uso

## *Instalación del MIDlet*

El paquete de instalación consta de una carpeta con dos ficheros:

- myEpideMobile.jad
- myEpideMobile.jar

Hay que proceder a copiarlos a una carpeta dentro del árbol de directorios del dispositivo móvil con el que se vaya a trabajar. La ruta de esta carpeta no es relevante.

Una vez copiados, hay que ejecutar cualquiera de los dos ficheros, y si el dispositivo móvil es compatible con el perfil MIDP 2.0 de Java ME se instalará sin problemas. Hay que tener en cuenta que el gestor de MIDlets administrará la aplicación, así que una vez instalada, se pueden borrar los dos ficheros que se han copiado al principio.

Normalmente el propio gestor permitirá la ejecución del programa, así como su borrado.

## *Configuración del MIDlet*

Antes de la instalación de la aplicación, hay que configurar los ficheros settings.txt y symptoms.txt que van dentro del fichero .jar.

El fichero symptoms.txt debe contener los síntomas que se van a tener en cuenta a la hora de recoger información epidemiológica, de tal forma de que haya un síntoma por línea.

El fichero settings.txt recoge los parámetros de configuración para que la aplicación recoja y pueda enviar los datos de manera apropiada. Cada línea debe contener un valor:

1. Código de operación
2. Código de base
3. Código de dispositivo
4. URL del clúster central
5. Variable que recogerá los datos en el clúster, seguida del signo =
6. Nombre del fichero donde se almacenarán los registros
7. Ruta local de la tarjeta de memoria



upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

8. Código de idioma

### *Ejecución del MIDlet*

Cuando se lanza la aplicación aparecerá un menú principal con las siguientes opciones:

Nuevo registro diario
Ver registro diario
Edita registro diario
Eliminar registro diario
Transferencia de datos
Configuración

Además, en el extremo izquierdo y derecho de la zona inferior de la pantalla se permite salir de la aplicación o entrar en una de las opciones del menú. Estos dos botones, además, permiten avanzar y retroceder en el resto de pantallas de la aplicación.

También se puede entrar en cualquiera de las opciones del menú con el botón central de las flechas de dirección del terminal, en el caso de que éste exista.

#### Nuevo registro diario

Esta opción permite crear un registro diario con la información epidemiológica recogida previamente.
La primera pantalla permite ajustar la fecha si el registro que se desea introducir no coincide con la fecha actual.
La segunda pantalla permite seleccionar el síntoma del que haya pacientes. *Sólo permite seleccionar uno solo, pero más adelante se podrá volver cuantas veces se desee para seleccionar otros y poder introducir más información.*
La tercera pantalla permite introducir el número de pacientes según sexo y edad.
La cuarta pantalla muestra, de manera esquemática, los datos introducidos hasta el momento.
En la quinta pantalla permite regresar a la pantalla de selección de síntomas o guardar los datos ya introducidos de ese día y regresar al menú principal. En caso de que se seleccione esta última opción, se generará un registro nuevo en un fichero de texto almacenado en el dispositivo móvil.



upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

### Ver registro diario

Esta opción permite visualizar los registros diarios con la información epidemiológica que hayamos introducido en el dispositivo previamente.
La primera pantalla muestra las fechas de los registros que hay almacenados para poder seleccionar el deseado.
La segunda y última pantalla muestra, de manera esquemática, el contenido del registro seleccionado.

### Editar registro diario

Esta opción permite editar un registro diario con la información epidemiológica que hayamos introducido en el dispositivo previamente.
La primera pantalla muestra las fechas de los registros que hay almacenados para poder seleccionar el deseado.
La segunda pantalla permite seleccionar el síntoma del que haya pacientes. *Sólo permite seleccionar uno solo, pero más adelante se podrá volver cuantas veces se desee para seleccionar otros y poder introducir más información.*
La tercera pantalla permite modificar el número de pacientes según sexo y edad.
La cuarta pantalla muestra, de manera esquemática, los datos introducidos hasta el momento.
En la quinta pantalla permite regresar a la pantalla de selección de síntomas para seguir editando o guardar los datos ya almacenados de ese día y regresar al menú principal. En caso de que se seleccione esta última opción, se modificará el registro ya existente en un fichero de texto almacenado en el dispositivo móvil.

### Eliminar registro diario

Esta opción permite eliminar los registros diarios con la información epidemiológica que hayamos introducido en el dispositivo previamente.
La primera pantalla muestra las fechas de los registros que hay almacenados para poder seleccionar el deseado.
La segunda pantalla muestra, de manera esquemática, el contenido del registro seleccionado. Darle al botón de continuar hace que se borre ese registro del fichero donde están almacenados, y se vuelve al menú principal.



upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

### Transferencia de datos

Esta opción permite transferir el archivo de registros almacenados en el dispositivo al clúster central o a la tarjeta de memoria, en caso de que ésta esté disponible.
Si se elige la primera opción, se establece una conexión con el servidor y envía el contenido del fichero a través del método POST. El fichero que contiene los datos será renombrado a oldrecords.txt.
La segunda opción hace que se copie la totalidad del fichero de registros a la tarjeta de memoria, y después se renombre el fichero del dispositivo a oldrecords.txt.
Después de seleccionar cualquiera de las opciones se vuelve al menú principal.

### Configuración

Esta opción debería permitir modificar los códigos de operación, base y dispositivo, pero en el estado actual de desarrollo de la aplicación no está activada esta funcionalidad.



upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

# Anexo IV: Licencia de uso

Este Programa se distribuye bajo la licencia GNU General Public License 2.0. Es una licencia de software libre. Como cualquier licencia de software libre, otorga las libertades siguientes:

1. La libertad de ejecutar el programa para cualquier propósito.
2. La libertad de estudiar cómo funciona el programa y adaptarlo a sus necesidades.
3. La libertad de redistribuir copias para ayudar al prójimo.
4. La libertad de mejorar el programa y divulgar las mejoras al público, para beneficio de toda la comunidad.

Se puede ejercer las libertades especificadas aquí siempre que cumpla con las condiciones expresadas en esta licencia. Las condiciones principales son:

- En cada copia que se distribuya, se debe difundir notoriamente y apropiadamente una nota referente a los derechos de autor y una exoneración de responsabilidades, y mantenerse intactos todos los avisos que se refieran a esta licencia y a la ausencia de cualquier garantía; y darse a cualquier destinatario del Programa una copia de la GNU General Public License junto con el Programa. Cualquier traducción de la GNU General Public License debe ir acompañada por la GNU General Public License.
- Si se modifica su copia o copias del programa o alguna parte de él, o se desarrolla un programa basado en él, se puede distribuir la obra resultante siempre bajo la GNU General Public License. Cualquier traducción de la GNU General Public License debe ir acompañada por la GNU General Public License.
- Si se copia o distribuye el programa, debe acompañarse del correspondiente código fuente leíble por máquinas o un ofrecimiento escrito, válido al menos por tres años, para subministrar el correspondiente código fuente completo.

A continuación, se muestra el contenido de la licencia. Se puede encontrar la traducción no oficial al español en http://www.gnu.org/licenses/old-licenses/gpl-2.0-translations.html



upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

**GNU GENERAL PUBLIC LICENSE**

Version 2, June 1991

Copyright (C) 1989, 1991 Free Software Foundation, Inc.
51 Franklin Street, Fifth Floor, Boston, MA 02110-1301, USA

Everyone is permitted to copy and distribute verbatim copies
of this license document, but changing it is not allowed.

## Preamble

The licenses for most software are designed to take away your freedom to share and change it. By contrast, the GNU General Public License is intended to guarantee your freedom to share and change free software--to make sure the software is free for all its users. This General Public License applies to most of the Free Software Foundation's software and to any other program whose authors commit to using it. (Some other Free Software Foundation software is covered by the GNU Lesser General Public License instead.) You can apply it to your programs, too.

When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for this service if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs; and that you know you can do these things.

To protect your rights, we need to make restrictions that forbid anyone to deny you these rights or to ask you to surrender the rights. These restrictions translate to certain responsibilities for you if you distribute copies of the software, or if you modify it.

For example, if you distribute copies of such a program, whether gratis or for a fee, you must give the recipients all the rights that you have. You must make sure that they, too, receive or can get the source code. And you must show them these terms so they know their rights.

We protect your rights with two steps: (1) copyright the software, and (2) offer you this license which gives you legal permission to copy, distribute and/or modify the software.

Also, for each author's protection and ours, we want to make certain that everyone understands that there is no warranty for this free software. If the software is modified by someone else and passed on, we want its recipients



upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

to know that what they have is not the original, so that any problems introduced by others will not reflect on the original authors' reputations.

Finally, any free program is threatened constantly by software patents. We wish to avoid the danger that redistributors of a free program will individually obtain patent licenses, in effect making the program proprietary. To prevent this, we have made it clear that any patent must be licensed for everyone's free use or not licensed at all.

The precise terms and conditions for copying, distribution and modification follow.

**TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION**

**0.** This License applies to any program or other work which contains a notice placed by the copyright holder saying it may be distributed under the terms of this General Public License. The "Program", below, refers to any such program or work, and a "work based on the Program" means either the Program or any derivative work under copyright law: that is to say, a work containing the Program or a portion of it, either verbatim or with modifications and/or translated into another language. (Hereinafter, translation is included without limitation in the term "modification".) Each licensee is addressed as "you".

Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running the Program is not restricted, and the output from the Program is covered only if its contents constitute a work based on the Program (independent of having been made by running the Program). Whether that is true depends on what the Program does.

**1.** You may copy and distribute verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and give any other recipients of the Program a copy of this License along with the Program.

You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

**2.** You may modify your copy or copies of the Program or any portion of it, thus forming a work based on the Program, and copy and distribute such



upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:

**a)** You must cause the modified files to carry prominent notices stating that you changed the files and the date of any change.
**b)** You must cause any work that you distribute or publish, that in whole or in part contains or is derived from the Program or any part thereof, to be licensed as a whole at no charge to all third parties under the terms of this License.
**c)** If the modified program normally reads commands interactively when run, you must cause it, when started running for such interactive use in the most ordinary way, to print or display an announcement including an appropriate copyright notice and a notice that there is no warranty (or else, saying that you provide a warranty) and that users may redistribute the program under these conditions, and telling the user how to view a copy of this License. (Exception: if the Program itself is interactive but does not normally print such an announcement, your work based on the Program is not required to print an announcement.)

These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Program, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based on the Program, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Program.

In addition, mere aggregation of another work not based on the Program with the Program (or with a work based on the Program) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.

**3.** You may copy and distribute the Program (or a work based on it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you also do one of the following:



upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

**a)** Accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,
**b)** Accompany it with a written offer, valid for at least three years, to give any third party, for a charge no more than your cost of physically performing source distribution, a complete machine-readable copy of the corresponding source code, to be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,
**c)** Accompany it with the information you received as to the offer to distribute corresponding source code. (This alternative is allowed only for noncommercial distribution and only if you received the program in object code or executable form with such an offer, in accord with Subsection b above.)

The source code for a work means the preferred form of the work for making modifications to it. For an executable work, complete source code means all the source code for all modules it contains, plus any associated interface definition files, plus the scripts used to control compilation and installation of the executable. However, as a special exception, the source code distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

If distribution of executable or object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place counts as distribution of the source code, even though third parties are not compelled to copy the source along with the object code.

**4.** You may not copy, modify, sublicense, or distribute the Program except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense or distribute the Program is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

**5.** You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Program or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Program (or any work based on the Program), you indicate your



upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Program or works based on it.

**6.** Each time you redistribute the Program (or any work based on the Program), the recipient automatically receives a license from the original licensor to copy, distribute or modify the Program subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties to this License.

**7.** If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Program at all. For example, if a patent license would not permit royalty-free redistribution of the Program by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Program.

If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to apply and the section as a whole is intended to apply in other circumstances.

It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system, which is implemented by public license practices. Many people have made generous contributions to the wide range of software distributed through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice.

This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.

**8.** If the distribution and/or use of the Program is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Program under this License may add an explicit geographical distribution limitation excluding those countries, so



upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.

**9.** The Free Software Foundation may publish revised and/or new versions of the General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program specifies a version number of this License which applies to it and "any later version", you have the option of following the terms and conditions either of that version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of this License, you may choose any version ever published by the Free Software Foundation.

**10.** If you wish to incorporate parts of the Program into other free programs whose distribution conditions are different, write to the author to ask for permission. For software which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

**NO WARRANTY**

**11.** BECAUSE THE PROGRAM IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

**12.** IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR



upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

**END OF TERMS AND CONDITIONS**



upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira