Desarrollo de aplicaciones para sistemas móviles con tecnología Symbian y Mobile Widgets

Javier García Puga [1]

[1] Telefónica Investigación y Desarrollo, Spain
jgarcia_puga@telefonica.es

Resumen: En este capitulo trabajaremos sobre el concepto de widget, que es una pequeña aplicación o programa, usualmente presentado en archivos o ficheros pequeños que son ejecutados por un motor de widgets o Widget Engine. Esta tecnología facilita el acceso a funciones frecuentemente usadas y proveer de información visual. Sin embargo, los widgets pueden hacer todo lo que la imaginación desee e interactuar con servicios e información distribuida en Internet; pueden ser vistosos relojes en pantalla, notas, calculadoras, calendarios, agendas, juegos, ventanas con información del clima en su ciudad, etcétera". En este capítulo se presenta esta tecnología y se muestran algunos ejemplos. Una vez analizados algunos ejemplos se mostrarán los diferentes tipos de widgets, clasificados en función del entorno de ejecución: widgets de escritorio (PC), widgets Web, mobile widgets, etc.

Palabras clave: Symbian; Mobile Widgets

Abstract. In this chapter we will work on the concept of widget, which is a small application or program, usually presented in files or small files that are executed by a widget engine or Widget Engine. This technology facilitates access to frequently used functions and provides visual information. However, widgets can do whatever the imagination desires and interact with services and information distributed on the Internet; they can be eye-catching on-screen clocks, notes, calculators, calendars, agendas, games, windows with information about the weather in your city, etc.". This chapter introduces this technology and shows some examples. Once some examples have been analysed, the different types of widgets will be shown, classified according to the execution environment: desktop widgets (PC), Web widgets, mobile widgets, etc.

Keywords: Symbian; Mobile Widget

ISBN: 978-84-9012-861-9
Colección Aquilafuente: AQ-241

1. Introdución

Según la wikipedia, un "*widget"* es una pequeña aplicación o programa, usualmente presentado en archivos o ficheros pequeños que son ejecutados por un motor de widgets o Widget Engine. Entre sus objetivos están los de dar fácil acceso a funciones frecuentemente usadas y proveer de información visual. Sin embargo, los widgets pueden hacer todo lo que la imaginación desee e interactuar con servicios e información distribuida en Internet; pueden ser vistosos relojes en pantalla, notas, calculadoras, calendarios, agendas, juegos, ventanas con información del clima en su ciudad, etcétera" [1-9].

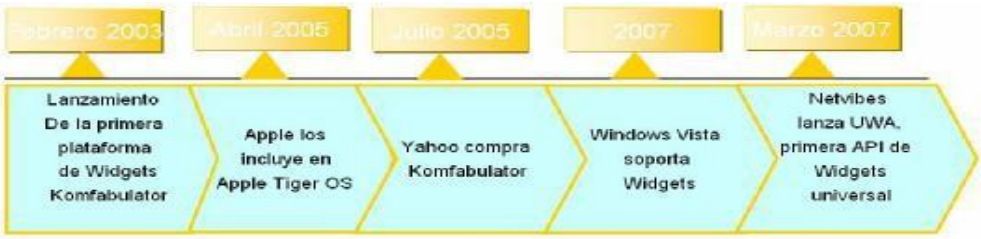
También son conocidos por términos equivalentes como "page components", "startlets", "gadgets", "web badges" o "modules", aunque la denominación más conocida es la de Widgets.

Existen diferentes tipos de widgets si se atiende al entorno donde éstos son ejecutados: widgets de escritorio (PC), widgets Web, mobile widgets, etc.

1.1 Características principales

- Son aplicaciones vistosas, ligeras y sencillas diseñadas para una finalidad concreta.
- Su aspecto es el de una aplicación independiente integrada dentro del escritorio
- Muy usables
- Gran nivel de acceso a los recursos de la máquina
- Fácil instalación / desinstalación
- Gran comunidad de desarrolladores
- Enmarcados dentro del movimiento Web 2.0 (AJAX, ...)

1.2 Evolución



Este tema se centrará en el desarrollo de widgets para terminales móviles (mobile widgets), especialmente en la plataforma Widsets y Nokia Web RunTime (WRT).

ISBN: 978-84-9012-861-9

Colección Aquilafuente: AQ-241

2. Widsets

2.1 Introducción

WidSets es una plataforma de widgets terminales móviles, basada en Java MIDP 2.0 y desarrollada por una *spin off* de Nokia.
La plataforma permite el desarrollo de widgets, utilizando el lenguaje de scripting Helium, que se ejecutarán en una máquina virtual (Widsets VM), la cual reside en el terminal móvil. Esta máquina virtual puede ser instalada en cualquier terminal que soporte Java MIDP 2.0 [10-15].

Las características principales de esta plataforma son las siguientes:
- No permite acceso a recursos del terminal (cámara, llamadas,…)
- Exige descarga de la plataforma (Widsets VM) y registro de usuarios en el terminal.
- El descubrimiento de widgets se puede realizar a través de web y se descarga en el terminal mediante sincronización.
- Permite creación de widgets por parte de usuarios mediante plantillas y el lenguaje de scripting Helium.
- Permite a los usuarios monitorizar el tráfico de que generan sus terminales.

Antes de empezar a enumerar los componentes de los widgets de esta plataforma conviene definir algunos conceptos:
- Gestor de WidSets (WidSets manager): Se trata de una aplicación web que proporciona a los usuarios de Widsets funcionalidades como la sincronización de widgets, gestión de la cuenta de usuario, monitorización del tráfico, etc.
- Biblioteca de WidSets (WidSets Library): Biblioteca publicada en un servidor donde residen los widgets una vez publicados.

2.2 omponentes de los widgets

Un widget está formado normalmente por los siguientes componentes:
- Fichero widget.xml. Este fichero contiene los detalles específicos del widget: meta-datos, servicios utilizados por el widget, hoja de estilo, recursos y layout de las vistas.
- Fichero script .he. Contiene los scripts de Helium utilizados para dotar de funcionalidad al widget.
- Ficheros de recursos:
  - web_icon.png: Imagen de 60*40px que será utilizada en la librería de widgets o en el estante (*shelf*) del Widgets Manager
    -

ISBN: 978-84-9012-861-9
Colección Aquilafuente: AQ-241
Ediciones Universidad de Salamanca - cc by nc dc

Figura 2.1 web_icon.png

- web_minimized.png: Esta imagen se utiliza para mostrar la vista minimizada del widget dentro del WidSets Manager de la web. El ancho de la imagen debe ser de 110 píxeles. La altura es flexible.

Figura 2.2 web_minimized.png

ISBN: 978-84-9012-861-9

Colección Aquilafuente: AQ-241
Ediciones Universidad de Salamanca - cc by nc dc

- web_maximized.png: Esta imagen se utiliza para mostrar la vista maximizada del widget dentro del WidSets Manager de la web. El tamaño de la imagen debe ser de 176x208px.

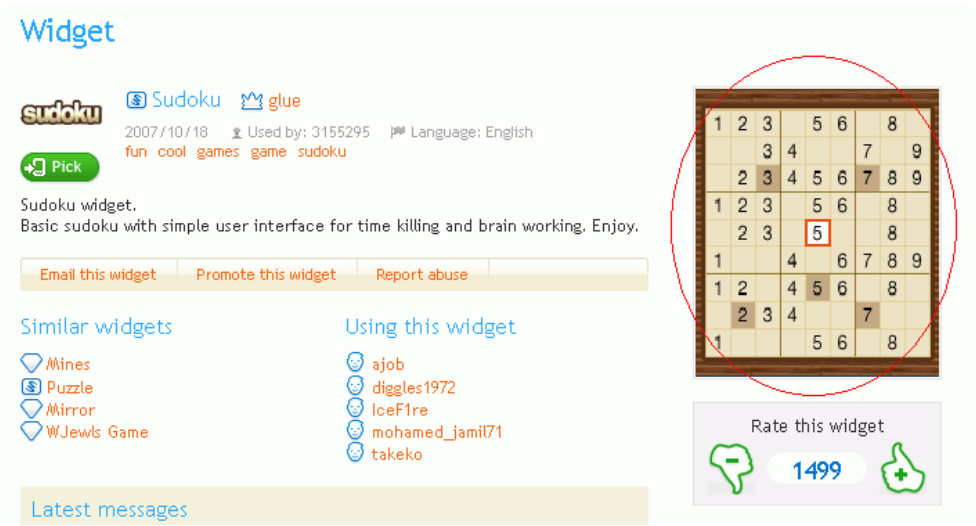


Figura 2.3 web_ maximized.png

2.2.1 Fichero widget.xml

El fichero *widget.xml* contiene todas las propiedades que definen el widget: nombre, versión, recursos, etc.
Este fichero debe comenzar con las siguientes líneas:

```
<?xml version="1.0" encoding="UTF-8"?>
  <widgetspec_version=”2.0”>
```

A continuación se debe incluir el elemento <info>, el cual contiene los siguientes elementos:

- name: Nombre del widget
- version: Versión
- author: Nombre del desarrollador.
- shortdescription: Descripción corta
- clientversion: (opcional) Versión del cliente que se necesita para ejecutar el widget
- longdescription: (opcional) Descripción corta
- tags: (opcional) Tags que describen el widget.

ISBN: 978-84-9012-861-9
Colección Aquilafuente: AQ-241
Ediciones Universidad de Salamanca - cc by nc dc

```
<info>
   <name>HolaMundo</name>
   <version>1.0</version>
   <author>Javier</author>
   <shortdescription>Descripcion   corta</shortdescription>
   <clientversion>0.97</clientversion>
   <longdescription> Descripción
      larga
   </longdescription>
   <tags>deportesnoticias</tags>
</info>
```

A parte del elemento *info* hay otros que simplemente se enumerarán a continuación:

- Help: Definir la ayuda del widget
- Services: Permiten realizar llamadas servicios ubicados en el servidor.
- Parameters: Los parámetros sirven para almacenar datos de configuración dinámicos.
- Variables: Variables del widget. Pueden hacer referencia a parámetros utilizando la sintaxis ${}
- Resources: Define los recursos del widget: imágenes, código, hoja de estilos, etc.
- Layout: Define las vistas y layout del widget.

2.3 Introducción al lenguaje de scripting Helium

Esta sección se enumerará de forma muy concisa las principales funcionalidades del lenguaje Helium. Como se observará, se trata de un lenguaje similar a Java.

Para más información http://dev.widsets.com/apidocs/

2.3.1 Elementos básicos

Comentarios

```
/* bloque */
// línea
```

Literales

```
”lorem ipsum” /* string */ ’\n’     /*
```

ISBN: 978-84-9012-861-9

Colección Aquilafuente: AQ-241

```
Caracter        */
123456          /* enter */
0777777         /* octal */
0xcafebabe      /* hexa */
0b101010101     /* binario*/
```

Control de flujo

```
if   else   while   do   for   switch   case   default
return   break   continue   foreach
```

Operadores

```
+    -    *    /    %    &    |    ^    <    >    !    ~
<<   >>   +=   -=   *=   /=   %=   &=   |=   ^=   <<=
<=   >=   ==   !=   &    ||   ++   --
instanceof      new ?:
```

Tipos primitivos

```
boolean /* true/false */
int         /* con signo 32 bit */ long
            /* sin signo 64 bit */
```

Keywords

```
class const false null return struct true void
```

2.3.2 Funciones más comunes

Las funcionalidades de los widgets pueden ser divididas en las siguientes categorías:

Interfaces relacionados con el ciclo de vida del widget

ISBN: 978-84-9012-861-9
Colección Aquilafuente: AQ-241
Ediciones Universidad de Salamanca - cc by nc dc

void startWidget(); //Llamada cuando el usuario inicia WidSets.
void stopWidget(); //Llamada cuando el usuario elimina o recarga el widget, o cuando se cierra WidSets .
Shell openWidget(); // Cuando se selecciona un widget para abrirlo (modo maximizado)
void closeWidget(); // Cuando se sale del modo maximizado del widget

Interfaces relacionados con menú de control del widget

MenuItem getSoftKey(Shell shell, Component focused, int key); //Llamada cuando el usuario pulsa una softKey
Menu getMenu(Shell shell, Component focused); //Llamada cuando el usuario pulsa una softKey asociada a un menú

Interfaces relacionados con las acciones de los usuarios

void actionPerformed(Shell shell, Component source, int action); //Llamada cuando el usuario pulsa una softKey
boolean keyAction(Component source, int op, int code); //Llamada cuando el usuario pulsa una tecla

Interface para la creación de la vista del widget.

Flow createView(String name, Object context)
Component createElement(String viewId, String elementId,
Style style, Object context);

Función de callback para eventos del timer

void timerEvent(Timer timer);

ISBN: 978-84-9012-861-9

Colección Aquilafuente: AQ-241

Funciones de callback para eventos del servidor de comunicaciones

void onSuccess(Object state, Value returnValue); void onFailure(Object state, String errorMessage);

2.3.3 Reglas del lenguaje

- Todas las funciones y variables deben estar implementadas siempre dentro de una clase.

```
class
{
   int globalVar = 100;

   void firstFunction()
   {
      int localVar = 0;
   }
}

//No obstante, pueden existir funciones dentro de otra función:


Flow createAddressView(String name, String addr, String city)
{
   Flow flow = new Flow(getStyle("address.view"));

   add("Name", name);
   add("Address", addr);
   add("City", city);

   void add(String name, String value)
   {
      String text = format("%s: %s", name, value); Label label =
      new Label("address.field"), text); label.setPreferred-
      Width(-100); label.setFlags(VISIBLE|LINEFEED);
      flow.add(label);
   }

   return flow;
}
```


ISBN: 978-84-9012-861-9
Colección Aquilafuente: AQ-241
Ediciones Universidad de Salamanca - cc by nc dc

- Todas las variables deben ser inicializadas y tienen alcance delimitado por el bloque en la que están contenidas.

```
voidfirstFunction()
{
   int j; /* error */
   {
      int c = 0;
   }
   {
      int d = c; /* error, 'c' no es visible aqui*/
   }
}
```

- Type-casting explícito

```
boolean flag = false; Text
displayText;

voidsomeFunction()
{
   int zeroOrOne = int(flag); /* casting boolean a int */
   ...
   String str = String(zeroOrOne); /* casting int a string */ displayText.setText(str);
}
```

2.3.4 *Limitaciones de Helium*

De momento, no muchas funcionalidades del terminal son accesibles a través del lenguaje Helium. A continuación se enumeran algunos ejemplos:

Acceso a memoria persistente del terminal.

Aunque no se puede acceder al sistema de ficheros del teléfono, se puede almacenar datos en memoria persistente utilizando la clase Store.

Acceso a Galería/Imágenes

N/
D

ISBN: 978-84-9012-861-9

Colección Aquilafuente: AQ-241

Acceso a contactos

N/
D

Bluetooth

N/
D

Envío recepción de SMS

N/
D

Realización de llamadas

N/
D

Lanzar el navegador.

De momento no está disponible, aunque es posible que en un futuro se implemente esta opción.

Reproducción de sonidos

Sí, utilizando la clase Player aunque no hay soporte para streaming.

Reproducción de videos

N/
D

ISBN: 978-84-9012-861-9
Colección Aquilafuente: AQ-241
Ediciones Universidad de Salamanca - cc by nc dc

2.4 Instalación de la plataforma de WidSets en el terminal

Para la instalación de la plataforma en un móvil es necesario realizarlo a través de su sitio web y realizar la descarga inicial de cómo mínimo un Widget, para que este instale de forma automática el motor de WidSets [16-20]. Es necesario realizar la creación de una cuenta de usuario con la que autenticarse en posteriores usos.

A continuación se indican los pasos a seguir para la instalación de la plataforma:

- Para instalar WidSets por primera vez en un móvil hay que visitar la web, www.widsets.com e ir a la sección "Get Started".
- A continuación se deberá marcar la casilla Pick que hay en cada Widget disponible (se marcarán las casillas Pick de tantos como se quiera instalar) y a continuación pulsar “Download the client and picked widgets”.
- Se completan los datos de registro y se eligen cualquiera de las opciones para transferir al móvil. La opción más cómoda es la de recibir la URL de descarga por SMS, para ello será necesario introducir el número de teléfono.
- En este momento un SMS es enviado al número de móvil introducido, con un link para descargar el motor de WidSets. En la página web se muestra un mensaje que lo confirma.
- Una vez que se recibe el SMS en el móvil hay que abrir el link de descarga.
- Una vez que la instalación ha finalizado, para ejecutar la aplicación hay que ir a la carpeta donde se ha guardado la aplicación, que dependiendo del teléfono ésta será Aplicaciones, Mis Cosas, etc.
- Pulsar sobre ”WidSets” y responder sí a la pregunta ¿Permitir a la aplicación WidSets usar la red y enviar y recibir datos?
- Al abrir la aplicación se informará de que es necesario registrarse en www.widsets.com, para ello hay que hacer lo siguiente:
- En el correo recibido pulsar en “Validate now”. Se valida el correo y se informa de que los procedimientos en caso de olvido de la password.
- Una vez completado el proceso de instalación de la plataforma, si se quieren añadir nuevos Widgets se puede realizar activando el Widget de Sistema desde el móvil. Se selecciona “library”, se marca la casilla Pick del Widget que se quiera instalar y seleccionar “Back”. Automáticamente el nuevo Widget se cargará en el móvil.
- La carga de nuevos Widgets en el móvil también se puede realizar desde la web www.widsets.com.

ISBN: 978-84-9012-861-9

Colección Aquilafuente: AQ-241

2.5 WidSets SDK

El SDK para WidSets, también conocido como DevKit, contiene todas las herramientas necesarias para poder desarrollar Widgets basados en esta tecnología [21-25]. El DevKit contiene un compilador de Helium, un emulador, ejemplos y documentación; y puede ser descargado desde la siguiente URL: http://dev.widsets.com/downloads/devkit.zip

El DevKit se utiliza una interfaz basada en comandos. Su sintaxis es la siguiente:

devkit <options> command <arguments>

Antes de poder utilizar el SDK, será necesario crear una cuenta en el site *http://dev.widsets.com*, puesto que la obtenida anteriormente en www.widsets.com no será válida.

Una vez creada la cuenta, se procederá a logarse en el sistema utilizando el siguiente comando:

devkit login <usuario> <clave>

Para lanzar el emulador se utilizará el comando:

devkit run

ISBN: 978-84-9012-861-9
Colección Aquilafuente: AQ-241
Ediciones Universidad de Salamanca - cc by nc dc

Figura 2.4 Emulador WidSets

Una vez tengamos desarrollada la aplicación, se podrá subir al site *dev.widsets.com* y verla en el emulador utilizando el comando: devkit upload <directorio proyecto>

2.6 Información de Referencia

- http://dev.widsets.com/wiki/Getting_started
- http://developer.widsets.com/apidocs

ISBN: 978-84-9012-861-9

Colección Aquilafuente: AQ-241

3. Nokia Web Run Time

3.1 Introducción

La plataforma S60, a partir de su 3ª edición FP 2, incorpora soporte para widgets por medio del Web Run-Time (WRT).
Los widgets se desarrollan de forma similar a los widgets web, utilizando JavaScript, CSS y HTML. WRT soporta las APIs de JavaScript de acuerdo con la 3ª edición de la especificación ECMA-262. También soporta JavaScript DOM y XMLHttpRequest. A parte del estándar, WRT proporciona APIs para permitir el acceso a varias propiedades y funcionalidades del terminal móvil [26-30].

Un widget debe estar compuesto como mínimo por dos ficheros, los cuales deben estar situados en el directorio raíz del proyecto. Además puede contener otros ficheros opcionales como un icono, ficheros css, ficheros javascript y otras imágenes.
La siguiente tabla resume los ficheros que componen un widget así como su ubicación relativa dentro del directorio donde se ubica el proyecto:

Fichero	Obl./Opc.	Ruta	Descripción
info.plist	Obligatorio	/	Fichero XML que contiene las propiedades del widget
[nombre].html	Obligatorio	/	Fichero HTML que contiene la estructura del widget. El nombre de este fichero debe venir especificado en el archivo info.plist
icon.png	Opcional	/	
*.css	Opcional	/ o cualquier subdirectorio	
*.js	Opcional	/ o cualquier subdirectorio	
*.jpg/bmp/gif/png	Opcional	/ o cualquier subdirectorio	

3.2 Fichero info.plist

Se trata de un fichero XML en el cual se definen las propiedades y las opciones de configuración del widget.
A continuación se muestra un ejemplo de la estructura de este fichero así como una tabla con la descripción de cada una de las propiedades que lo componen:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE plist PUBLIC "-//Nokia//DTD PLIST 1.0//EN"
"http://www.nokia.com/NOKIA_COM_1/DTDs/plist-1.0.dtd">
<plist version="1.0">
<dict
```


ISBN: 978-84-9012-861-9
Colección Aquilafuente: AQ-241

```
        </dict>
<key>DisplayName</key>
<string>Nombre del Widget</string>
<key>Identifier</key>
<string>com.company.widget.name</string>
<key>MainHTML</key>
<string>Main.html</string>
<key>Version</key>
<string>1.0</string>
<key>AllowNetworkAccess</key>
<false/>
        </plist>
```

Nombre	Tipo	Obl./Opc.	Descripción
DisplayName	String	Obligatorio	Nombre que aparecerá en el menú de aplicaciones
Identifier	String	Obligatorio	Identificador único del widget
MainHTML	String	Obligatorio	Nombre de la página HTML que será cargada cuando se ejecute el widget.
Version	String	Opcional	Versión del widget
AllowNetworkAccess	Bolean	Opcional	Indica si el widget requiere acceso a la red.

3.3 Fichero HTML

Un widget debe tener un fichero HTML en el cual se definen su apariencia y funcionalidad. Se recomienda seguir la especificación HTML 4.01 a la hora de implementar este fichero.

El nombre de este fichero deberá coincidir con lo especificado en la propiedad MainHTML del fichero info.plist.

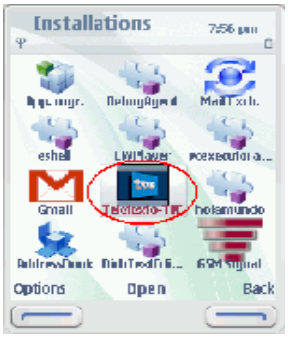
Durante la ejecución del widget, este documento HTML no podrá ser sustituido por otro archivo HTML. Si un widget necesita mostrar una página HTML adicional, esta página deberá mostrarse directamente desde el browser web. Para ello se podrá utilizar la función *widget.openURL()* analizada más adelante.

3.4 icon.png

El fichero icon.png será el que utilice el sistema operativo a la hora de mostrar el widget dentro del menú de aplicaciones. Deberá ser una imagen de 88x88 píxeles con formato png.

ISBN: 978-84-9012-861-9

Colección Aquilafuente: AQ-241



En el caso que no se incluya este fichero, el sistema operativo asignará un icono por defecto a widget.

3.5 Ficheros .css

El fichero CSS (u hoja de estilo) sirve para configurar el layout y estilo del widget. Symbian acepta cualquier fichero siga la especificación CSS nivel 2 o 3.
Aunque el estilo del widget puede estar contenido dentro del fichero Main HTML, utilizando los atributos *style* dentro de los tags html, se recomienda la utilización del fichero CSS de cara a separar la apariencia del modelo de datos. Un widget puede tener tantos ficheros CSS como se necesiten [31-36].
Para enlazar una CSS a un fichero HTML se podrá utilizar cualquiera de los tags de HTML
utilizados para tal
efecto:

```
<link rel="stylesheet" type="text/css" href="Stylesheet.css" />
```

También es posible enlazar ficheros CSS que se encuentren en un servidor remoto. En este caso el valor de la propiedad *AllowNetworkAccess* del fichero info.plist deberá ser *true*.

```
<link rel="stylesheet" type="text/css" href="http://myserver.com/Stylesheet.css"/>
```

3.6 Ficheros Javascript

Los ficheros .js contienen el código JavaScript que contendrá la lógica del widget. Aunque el código JavaScript puede estar contenido dentro del fichero HTML, se recomienda la utilización de ficheros .js por motivos de modularidad y reutilización de código. Un widget puede tener tantos ficheros .js como se necesiten.

```
<script type='text/JavaScript' src='JavaScript.js'></script>
```

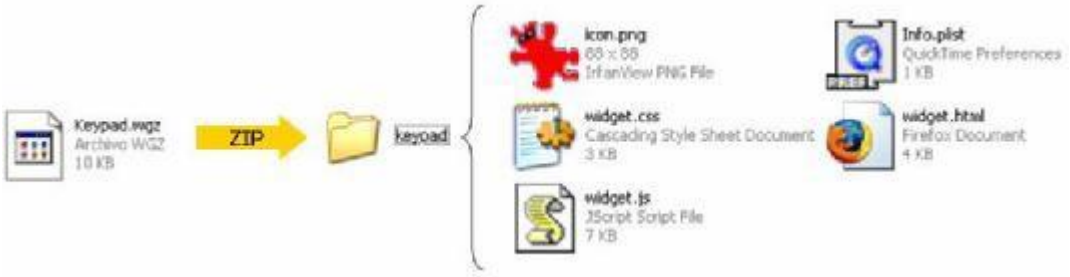
Al igual que en el caso anterior, también es posible enlazar ficheros .js que se encuentren en un servidor remoto. En este caso el valor de la propiedad *AllowNetworkAccess* del fichero info.plist deberá ser *true*.

```
<script type='text/JavaScript' src='http://www.widget.server/JavaScript.js'></script>
```


ISBN: 978-84-9012-861-9
Colección Aquilafuente: AQ-241
Ediciones Universidad de Salamanca - cc by nc dc

3.7 Preparación paquete del paquete de instalación

Antes de poder instalar el *widget* en el terminal es necesario generar el paquete de instalación. Este proceso es muy sencillo y consiste en comprimir con ZIP la carpeta donde está contenido el proyecto. A continuación deberá renombrarse la extensión del archivo a *.wgz.*



3.8 Instalación del widget

Los widgets pueden ser desplegados en los terminales móviles por varias formas:

- Por bluetooth o infrarrojos.
- Por medio del puerto USB o copiándolo directamente en la tarjeta multimedia MMC
- A través del navegador web, siempre que el servidor tenga configurado el tipo mime *x- nokia-widget* para los ficheros con extensión *.wgz*

3.9 Desarrollo en PC

Los widgets basados en WRT pueden ser probados y depurados en un PC, utilizando Firefox y algunos de sus extensiones:

- Extensión Firebug para depurar el widget.
- Extensión GreaseMonkey junto con el script "XmlHttpRequest Bypass Security", para eliminar alguna restricción de seguridad del objeto XmlHttpRequest

Por otra parte, existe un JavaScript (en desarrllo) en el cual están definidos algunos de los objetos JavaScript propietarios utilizados en WRT (widget, menú, etc). Este JavaScript se puede descargar de:
http://wiki.forum.nokia.com/images/e/e5/Widget_desktop_21112007.zip

ISBN: 978-84-9012-861-9

Colección Aquilafuente: AQ-241

3.10 API JavaScript

En el siguiente apartado se expone la API de JavaScript que pueden utilizar los desarrolladores a la hora de dotar de funcionalidad el widget [37-40].
A parte de los objetos y clases proporcionadas por el estándar JavaScript, el WRT incorpora los siguientes nuevos objetos:

- widget
- menu
- MenuItem
- SystemInfo

3.11 API JavaScript: Objeto widget

El objeto widget dispone de los siguientes métodos y propiedades:

- Métodos:
  - void openURL(String url)
  - void setPreferenceForKey (String preference, String key)
  - String preferenceForKey (String key)
  - void prepareForTransition(String transitionState)
  - void performTransition(void)
  - void setNavigationEnabled(Boolean flag)
  - void openApplication(HexNumber Uid, String param)
  - void setDisplayLandscape(void)
  - void setDisplayPortrait(void)

- Propiedades:
  - String identifier
  - void [Function] onshow
  - void [Function] onhide
  - Boolean isrotationsupported

3.11.1 *void openURL(String url)*

Syntaxis:

- [void] window.widget.openURL(String url)
- [void] widget.openURL(String url)

Descripción

- Abre la url especificado en el browser del terminal. El widget permanecerá abierto aunque en segundo plano.

ISBN: 978-84-9012-861-9
Colección Aquilafuente: AQ-241
Ediciones Universidad de Salamanca - cc by nc dc

Parámetros de entrada:

- url: La url que se desea abrir en el navegador.

Valor de retorno:

- Ninguno

Ejemplo:

- widget.openURL("www.google.es");

3.11.2 setPreferenceForKey()

Syntaxis:

- [void] window.widget.setPreferenceForKey(String preference, String key)

DescripciónPermite almacenar un par clave/valor de forma persistente. Las claves únicamente son eliminadas del terminal en caso de que el widget sea desinstalado. También se puede eliminar una clave especificando como valor "null"

Parámetros de entrada:

- preference: valor
- key: clave

Valor de retorno:

- Ninguno

Ejemplo:

```
var valor = document.forms[0].input1.value; widget.setPreferenceForKey('input1',
valor);
```

ISBN: 978-84-9012-861-9

Colección Aquilafuente: AQ-241

3.11.3 preferenceForKey()

Syntaxis:

- [string] window.widget.preferenceForKey(String key)

Descripción

- Permite obtener un valor almacenado previamente por la función *setPreferenceForKey()*

Parámetros de entrada:

- Clave del valor

Valor de retorno:

- Cadena conteniendo el valor o *undefined* en caso de que la clave no exista.

Ejemplo:

```
if(widget.preferenceForKey('input1'))
{
var valor = widget.preferenceForKey('input1'); document.forms[0].input1.value =
valor;
}
```

3.11.4 prepareForTransition()

Syntaxis:

- [void] widget.prepareForTransition(String transitionMode)

Descripción

- Este método se utiliza conjuntamente con performTransition y su finalidad es preparar al widget antes de modificar su apariencia, para prevenir posibles parpadeos (*flickering*).
- Por ello es conveniente llamarlo antes de utilizar las siguientes propiedades de los objetos que componen la vista:
  - [object].style.display = “block”: Para hacer visible un elemento.

ISBN: 978-84-9012-861-9
Colección Aquilafuente: AQ-241
Ediciones Universidad de Salamanca - cc by nc dc

- [object].style.display = “none”: Para ocultarlo.

Parámetros de entrada:

- transitionMode: Cadena que define el modo de transición deseado. Actualmente solo está soportado el modo "fade" que provoca que la apariencia del widget se modifique con un efecto de desvanecimiento [41-45].

Valor de retorno:

- Ninguno

3.11.5 performTransition()

Syntaxis:

- [void] widget.performTransition(void)

Descripción

- Este comando se tiene que utilizar para actualizar la pantalla, una vez que se ha llamado al método prepareForTransition y se han realizado los cambios pertinentes en la vista.

Parámetros de entrada:

- Ninguno

Valor de retorno:

- Ninguno

Ejemplo:

Fichero .js

```
function toMain(main) { widget.prepareForTransi-
        tion("fade"); if (main) {
        document.getElementById("config").style.display = 'none';
        document.getElementById("main").style.display='block';
```

ISBN: 978-84-9012-861-9

Colección Aquilafuente: AQ-241

```
        }
        else {
        document.getElementById("main").style.display = 'none'; docu-
        ment.getElementById("config").style.display = 'block';
        }
        widget.performTransition();
}
```

Main.html

```
<html>
<head>
        <title>Widget</title>
        <script type='text/JavaScript' src='holamundo.js'></script>
</head>

<body>
<div id='main'> Hola
Mundo !!
<input type="button" value="AcercaDe" onclick="toMain(0);" />
</div>
<div id='config' style="display:none">
<img src="usal.jpg" alt=""/>
<br/>
Universidad de Salamanca
<input type="button" value="Volver" onclick="toMain(1);" />
</div>
</body>
</html>
```

3.11.6 setNavigationEnabled()

Syntaxis:

- window.widget.setNavigationEnabled(Boolean navigationMode)

Descripción

- Este método se utiliza para cambiar la forma de navegar por el widget. Se puede elegir entre navegación con el cursor o tabulada.

ISBN: 978-84-9012-861-9
Colección Aquilafuente: AQ-241
Ediciones Universidad de Salamanca - cc by nc dc

Parámetros de entrada:

- navigationMode: Si es true: modo navegación por cursor. Si es false: Modo navegación tabulada.

Valor de retorno:

- Ninguno

Ejemplo:

```
// Modo Tab widget.setNavigationEna-
bled(false);
// Modo Cursor
widget.setNavigationEnabled(true);
```

La siguiente figura representa los dos modos de navegación, por cursor y tabulado:

Si el modo de navegación es tabulado (*navigationMode=false*), se pueden capturar eventos del teclado (*keyevents*) utilizando JavaScript:

ISBN: 978-84-9012-861-9

Colección Aquilafuente: AQ-241

```
document.onkeydown = keyDown; func-
tion keyDown(event) {
    if(event.keyCode==49) { //“1”
```

La siguiente figura recoge los códigos de tecla que se pueden capturar:

ISBN: 978-84-9012-861-9
Colección Aquilafuente: AQ-241
Ediciones Universidad de Salamanca - cc by nc dc

Key	Key press	Key down	Key up
0	48/48	48/48	48/48
1	49/49	49/49	49/49
2	50/50	50/50	50/50
3	51/51	51/51	51/51
4	52/52	52/52	52/52
5	53/53	53/53	53/53
6	54/54	54/54	54/54
7	55/55	55/55	55/55
8	56/56	56/56	56/56
9	57/57	57/57	57/57
*	56/42	42/42	56/42
#	51/35	35/35	51/35
C (del)	8/8	8/8	8/8
Call creation key (green)	0/63586	63586/63586	0/63586
Center	0/63557	63557/63557	n/a
Left	37/63495	63495/63495	n/a
Up	38/63497	63497/63497	n/a
Right	39/63496	63496/63496	n/a
Down	40/63498	63498/63498	n/a

3.11.7 openApplication()

Syntaxis:

- [void] window.widget.openApplication(HexNumber Uid, String param)

Descripción

- Lanza una aplicación nativa (S60 C++) en modo stand-alone.

Parámetros de entrada:

- Uid: UID de la aplicación nativa
- param: Cadena que especifica el argumento pasar a la aplicación nativa

ISBN: 978-84-9012-861-9

Colección Aquilafuente: AQ-241

Valor de retorno:

- Ninguno

Ejemplo:

```
// Lanza la aplicación de contactos widget.openApplication(0x101f4cce,"");
```

3.11.8 setDisplayLandscape() / setDisplayPortrait()

Syntaxis:

- widget.setDisplayLandscape(void)
- widget.setDisplayPortrait(void)

Descripción

- Estos métodos se utilizan para cambiar la orientación de la pantalla del terminal.
  Se puede elegir entre modo retrato (setDisplayPortrait)o apaisado (setDisplayLandscape).

Parámetros de entrada:

- Ninguno

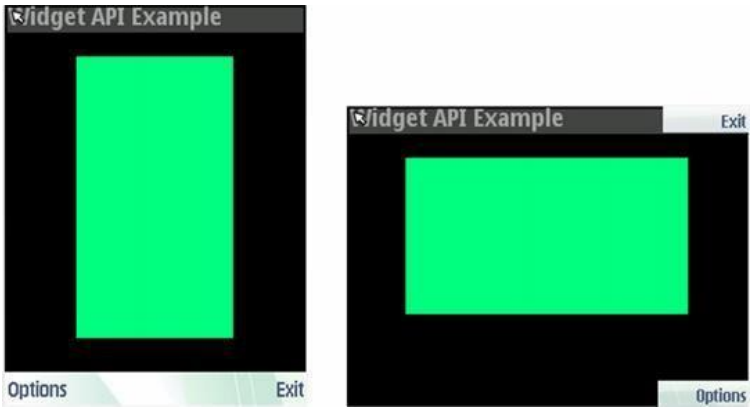
Valor de retorno:

- Ninguno

Ejemplo:

```
if(widget.isrotationsupported) widget.setDisplay-
        Landscape();
```


ISBN: 978-84-9012-861-9
Colección Aquilafuente: AQ-241
Ediciones Universidad de Salamanca - cc by nc dc



3.11.9 identifier

Syntaxis:

- [String] widget.identifier

Descripción

- Propiedad de solo lectura. Valor único que identifica a un widget, según está definido en la entrada "Identifier" del fichero info.plist.

3.11.10 onshow

Syntaxis:

- [void] widget.onshow = function() { }

Descripción

- Función de callback que será llamada cuando un widget pase de segundo plano a primer plano.

3.11.11 onhide

Syntaxis:

- [void] widget.onhide = function() { }

ISBN: 978-84-9012-861-9
Colección Aquilafuente: AQ-241
Ediciones Universidad de Salamanca - cc by nc dc

Descripción

- Función de callback que será llamada cuando un widget pase de primer plano a segundo plano.

3.11.12 isrotationsupported

Syntaxis:

- [Boolean] widget.isrotationsupported

Descripción

- Propiedad de solo lectura que retorna un valor booleano indicando si el terminal soporta orientación de pantalla apaisada o retrato. Si el valor es true, el terminal soporta ambas orientaciones de pantalla [46-50].

ISBN: 978-84-9012-861-9
Colección Aquilafuente: AQ-241
Ediciones Universidad de Salamanca - cc by nc dc

3.12 API JavaScript: Objeto menu

El objeto menu, junto con el objeto MenuItem, se utiliza para gestionar el menú de opciones del widget y las soft-keys. Este objeto dispone de los siguientes métodos y propiedades:

- Métodos:
  - void append(MenuItem menuItem)
  - void remove(MenuItem menuItem)
  - Object MenuItem getMenuItemById(Integer menuItemId)
  - Object MenuItem getMenuItemByName(String menuItemLabel)
  - void setRightSoftkeyLebel(String label, Function callbackfunction)
  - void showSoftkeys()
  - void hideSoftkeys()
  - void clear()
- Propiedades:
  - void [Function] onshow

3.12.1 *append()*

Syntaxis:

- [void] menu.append(MenuItem menuItem)

Descripción

- Este método permite añadir un ítem en el menú de opciones. Los ítems se muestran dentro del menú siguiendo el orden en el que son añadidos.

Parámetros de entrada:

- menuItem: instancia del objeto MenuItem que desea ser añadido en el menú de opciones.

Valor de retorno:

- Ninguno

3.12.2 *remove()*

ISBN: 978-84-9012-861-9

Colección Aquilafuente: AQ-241

Syntaxis:

- menu.remove(MenuItem menuItem)

Descripción

- Elimina una entrada del menú de opciones.

Parámetros de entrada:

- menuItem: instancia del objeto MenuItem que desea ser eliminado del menú de opciones.

Valor de retorno:

- Ninguno

3.12.3 getMenuItemById()

Syntaxis:

- [MenuItem] menu.getMenuItemById(Integer id)

Descripción

- Permite recuperar el handle a una instancia de un ítem de menú específico a partir de su id.

Parámetros de entrada:

- id: El identificador de un ítem de menú existente.

Valor de retorno:

- Instancia del objeto MenuItem. Si el id especificado es inválido, el método retorna "undefined".

3.12.4 getMenuItemByName()

ISBN: 978-84-9012-861-9
Colección Aquilafuente: AQ-241
Ediciones Universidad de Salamanca - cc by nc dc

Syntaxis:

- [MenuItem] menu.getMenuItemByName(String menuItemLabel)

Descripción

- Permite recuperar el handle a una instancia de un ítem de menú específico a partir de su nombre (label)

Parámetros de entrada:

- menuItemLabel: Nombre del ítem de menú que desea ser recuperado.

Valor de retorno:

- Instancia del objeto MenuItem. Si el nombre especificado es inválido, el método retorna "undefined".

Ejemplo:

3.12.5 setRightSoftkeyLabel()

Syntaxis:

- menu.setRightSoftkeyLabel(String label, Function callbackfunc)

Descripción

- Este método permite personalizar el nombre y la acción asociada a la softkey derecha. Por defecto, esta softkey está asignada a la función "Salir", la cual termina la ejecución del widget.
- Para reestablecer la softkey por defecto, se deberá llamar al método pasándole los

siguientes parámetros: menu.setRightSoftkeyLabel('', null)

ISBN: 978-84-9012-861-9

Colección Aquilafuente: AQ-241

Parámetros de entrada:

- label: Cadena de texto que especifica el título de la softkey
- callbackfunc: Referencia a la función de callback, la cual será llamada cuando se presione la softkey derecha.

Valor de retorno:

- Ninguno

Ejemplo:

```
function switchToSettingsView()
{
window.menu.setRightSoftkeyLabel('Back',   returnToMainView);
}
functionreturnToMainView()
{
// Cambiar la vista
...
// Reestablecer la softkey por defecto window.menu.setRightSoftkeyLabel('',
null);
}
```

3.12.6 showSoftkeys()

Syntaxis:

- [void] menu.showSoftkeys(void)

Descripción

- Muestra el panel inferior que contiene las softkeys, el cual está oculto por defecto.

Parámetros de entrada:

- Ninguno

ISBN: 978-84-9012-861-9
Colección Aquilafuente: AQ-241
Ediciones Universidad de Salamanca - cc by nc dc

Valor de retorno:

- Ninguno

Ejemplo:

```
window.menu.showSoftkeys();
```

3.12.7 hideSoftkeys()

Syntaxis:

- [void] menu.hideSoftkeys(void)

Descripción

- Oculta el panel que contiene las *softkeys*,

Parámetros de entrada:

- Ninguno

Valor de retorno:

- Ninguno

Ejemplo:

```
functionuseFullScreen()
{
window.menu.hideSoftkeys();
...
}
```

3.12.8 clear()

Syntaxis:

- [void] menu.clear(void)

ISBN: 978-84-9012-861-9

Colección Aquilafuente: AQ-241

Descripción

- Elimina todos los ítems del menú de opciones.

Parámetros de entrada:

- Ninguno

Valor de retorno:

- Ninguno

ISBN: 978-84-9012-861-9
Colección Aquilafuente: AQ-241
Ediciones Universidad de Salamanca - cc by nc dc

3.12.9 Propiedad: onShow

Syntaxis:

- [void] menu.onShow = menupaneShown;

Descripción

- Función de callback que será llamada cuando el menú de opciones es abierto, es decir, cuando el usuario presione el soft-button izquierdo.

3.13 API JavaScript: Objeto MenuItem

El objeto **MenuItem** permite gestionar, junto con el objeto menu, el menú principal del widget así como sus submenús [51].
El objeto MenuItem se crea utilizando el operador *new* y dispone de los siguientes métodos y propiedades:

- Métodos:
  - Object MenuItem(String label, Integer Id) o void append(MenuItem childMenuItem) o void remove(MenuItem childMenuItem) o void setDimmed(Boolean true|false)
- Propiedades:
  - void [Function] onSelect

3.13.1 Constructor

Syntaxis:

- [MenuItem] new MenuItem(String label, Integer id)

Descripción

- Se utiliza el operador new para crear una instancia del objeto.

Parámetros de entrada:

- label: Nombre del ítem
- id: Identificador único del ítem

ISBN: 978-84-9012-861-9

Colección Aquilafuente: AQ-241

Valor de retorno:

- Instancia del objeto MenuItem.

3.13.2 append()

Syntaxis:

- [void] MenuItem.append(MenuItem childMenuItem)

Descripción

- Este método permite crear submenús dentro del menú de opciones. Los ítems se muestran dentro del menú siguiendo el orden en el que son añadidos [52].

Parámetros de entrada:

- childMenuItem: instancia del objeto MenuItem que desea ser añadido en el menú padre.

Valor de retorno:

- Ninguno

3.13.3 remove()

Syntaxis:

- [void] MenuItem.remove(MenuItem childMenuItem)

Descripción

- Elimina una entrada (y sus hijos) de un menú padre.

Parámetros de entrada:

- childMenuItem: instancia del objeto MenuItem que desea ser eliminado del menú.

ISBN: 978-84-9012-861-9
Colección Aquilafuente: AQ-241
Ediciones Universidad de Salamanca - cc by nc dc

Valor de retorno:

- Ninguno

3.13.4 setDimmed()

Syntaxis:

- [void] MenuItem.setDimmed(Boolean flag)

Descripción

- Este método se utiliza para mostrar u ocultar un ítem de menú existente. Por defecto, un ítem de menú se muestra cuando se añade al menú de opciones.

Parámetros de entrada:

- flag: true para mostrar el ítem, false para ocultarlo.

Valor de retorno:

- Ninguno

3.13.5 Propiedad onSelect

Syntaxis:

- MenuItem.onSelect = function(Integer id) { }

Descripción

- Función de callback que es llamada cada vez que un usuario selecciona un ítem del menu. El argumento que se le pasa a esta función es el id del ítem que se ha seleccionado.
- También es posible asignar una función de callback independiente para cada ítem.
  En este caso, el parámetro id pasado a la función de callback puede ser ignorado.
- La función de callback debe ser asignada después de que el menú se haya añadido al menú principal.

ISBN: 978-84-9012-861-9

Colección Aquilafuente: AQ-241
Ediciones Universidad de Salamanca - cc by nc dc

Ejemplo:

Creación del menú

```
window.onload = createMenu();
function createMenu()
{
var optionsMenu = window.menu;

optionsMenu.onShow = function()
{
alert('Evento: optionsMenu.onShow');
}
// Crear 2 items
var m1 = new MenuItem('Beverages', 2001); var m2 =
new MenuItem('Snacks', 2002);
// Asignar la function de callback m1.onSelect =
menuEventHandler; m2.onSelect = menuE-
ventHandler;

optionsMenu.append(m1); optionsMenu.append(m2);
var m11 = new MenuItem('Coca Cola', 3001);
var m12 = new MenuItem('Pepsi', 3002); optionsMenu.getMenu-
ItemById(2001).append(m11); optionsMenu.getMenuItemByName('Beverag-
es').append(m12); m11.onSelect = submenuEventHandler;
m12.onSelect = submenuEventHandler;
}
```

Implementación del manejador de eventos:

```
function menuEventHandler(id)
{
switch (id)
{
case 2001:
break; case
2002:
// TODO
break;
}
}
```


ISBN: 978-84-9012-861-9
Colección Aquilafuente: AQ-241
Ediciones Universidad de Salamanca - cc by nc dc

3.14 API JavaScript: SystemInfo API

Este API permite al widget acceder a ciertas propiedades del terminal así como controlar algunas de sus funcionalidades [53].
El API está implementado como un plug-in, y por tanto debe ser cargado añadiendo el siguiente código en el fichero html principal:

<embedtype="application/x-systeminfo-widget"hidden="yes"></embed> La refe-

rencia al objeto puede ser recuperada vía JavaScript utilizando el siguiente código:

var sysinfo = document.embeds[0];

Los servicios proporcionados por este plug-in están divididos en las siguientes categorías, conteniendo cada una los métodos y propiedades que siguen a continuación:

- Información de la batería
  - Integer chargelevel
  - String [Function] onchargelevel
  - Boolean chargerconnected
  - String [Function] onchargerconnected
- Información de la red
  - Integer signalbars
  - String networkname
  - Integer networkregistrationstatus
- Información y gestión de la luz de la pantalla y teclado del dispositivo.
  - Const int lightminintensity
  - Const int lightmaxintensity
  - Const int lightdefaultintensity
  - Const int lightinfiniteduration
  - Integer lightmaxduration
  - Integer lightdefaultcycletime
  - Const int lighttargetprimarydisplayandkeyboard

ISBN: 978-84-9012-861-9

Colección Aquilafuente: AQ-241

    - Const int lighttargetsystem
    - Void lighton(Int lighttarget, Int duration, Int intensity, Bool fadein);
    - Void lightoff(String lighttarget, Int duration, Int fadeout);
    - Void lightblink(String lighttarget, Int duration, Int onduration, Int offduration, Int intensity);
- Información y gestión del vibrador.
    - Const integer vibramaxintensity o Const integer vibramaxduration o Const integer Vibrasettings
    - Void startvibra(Int duration, Int intensity);
    - Void stopvibra();
- Gestión de tonos.
    - Void beep(Integer frequency, Integer duration);
- Información de la memoria y sistema de ficheros.
    - Integer totalram o Integer freeram o String drivelist
    - Integer drivesize(String drivename)
    - Integer drivefree(String drivename)
- Información del idioma.
    - String language

3.14.1 chargelevel

Syntaxis:

- [int] sysinfo.chargelevel;

Descripción

- Propiedad de solo lectura que retorna un valor entero de entre 0 a 100 que indica el valor actual de batería en %.

3.14.2 onchargelevel

Syntaxis:

ISBN: 978-84-9012-861-9
Colección Aquilafuente: AQ-241
Ediciones Universidad de Salamanca - cc by nc dc

- sysinfo.onchargelevel = "batteryLevelChange()";
- function batteryLevelChange() { ...};

Descripción

- Esta propiedad es el gestor del evento que se produce cuando cambia el nivel de batería.

Ejemplo:

```
window.onload = function(){
var batteryLevel = sysinfo.chargelevel; alert("Nivel de batería:
" + batteryLevel + "%");

sysinfo.onchargelevel = "batteryLevelChange()";
}
function batteryLevelChange()
{
var batteryLevel = sysinfo.chargelevel; alert("Nivel de bat-
ería:" + batteryLevel + "%");
}
```

3.14.3 chargerconnected

Syntaxis:

- [boolean] sysinfo.chargerconnected;

Descripción

- Propiedad de solo lectura que retorna un booleano indicando el estado del cargador.
- Si el cargador está conectado devolverá true, si no false.

3.14.4 onchargerconnected

Syntaxis:

- sysinfo.onchargerconnected = "chargerConnectedEventHandler()";
- function chargerConnectedEventHandler(){…}

ISBN: 978-84-9012-861-9

Colección Aquilafuente: AQ-241
Ediciones Universidad de Salamanca - cc by nc dc

Descripción

- Esta propiedad es el gestor del evento que se produce cuando el cargador se conecta o desconecta al terminal.
- Debido a que el evento se lanza antes de actualizar el valor de la propiedad chargerconnected, se recomienda aplicar un pequeño retardo antes de leer dicho valor.

Ejemplo:

```
window.onload = function(){
sysinfo.onchargerconnected="chargerConnectedEvent()";
}
functionchargerConnectedEvent()
{
setTimeout("readValue();", 500);
}
function readValue()
{
if(sysinfo.chargerconnected) alert("El carga-
dor está conectado "); else
alert(“El cargador no está conectado”);
}
```

3.14.5 signalbars

Syntaxis:

- [int] sysinfo.signalbars;

Descripción

- Propiedad de solo lectura que retorna un entero (de 0 a 7) indicando el estado de la señal de red.
- 0 indica que no hay señal y 7 que la señal es máxima

Ejemplo:

```
var signalStrength = sysinfo.signalbars; if (signal-
Strength == 0)
alert("¡Sinseñal!");
```


ISBN: 978-84-9012-861-9
Colección Aquilafuente: AQ-241
Ediciones Universidad de Salamanca - cc by nc dc

3.14.6 networkname

Syntaxis:

- [string] sysinfo.networkname;

Descripción

- Propiedad de solo lectura que devuelve una cadena conteniendo el nombre de la red móvil registrada en ese momento por el terminal.

3.14.7 networkregistrationstatus

Syntaxis:

- [int] sysinfo.networkregistrationstatus;

Descripción

- Propiedad que retorna un entero (de 0 a 7) indicando el estado de registro de la red.
  0. Desconocido
  1. No registrado. El terminal no puede detectar otras redes.
  2. No registrado. El terminal puede detectar otras redes en las únicamente se puede realizar una llamada de emergencia.
  3. No registrado, pero el terminal está buscando a otra red con registrarse. operador de red. El terminal puede detectar otras redes cuales únicamente se puede realizar una llamada de emergencia.
  4. Registrado, red ocupada
  5. Registrado en red local
  6. Registro denegado
  7. Registrado en red visitante (roaming)

Ejemplo:

```
window.onload = function(){ net-
workRegEventHandler();
}
networkRegEventHandler() { var
textInfo = null;
var networkName = "";
var status = sysinfo.networkregistrationstatus; switch (status) {
case 0:
case 1:
```

ISBN: 978-84-9012-861-9

Colección Aquilafuente: AQ-241

```
case 2:
case 3:
case 4:
case 6:
textInfo = "Fallo en el registro de red"; break;
case 5:
networkName = sysinfo.networkname; textInfo
="Registrado en la red local "; break;
case 7:
networkName = sysinfo.networkname;
textInfo =" Registrado en la red visitante "; break;
default:
break;
}
alert(textInfo + networkName);
}
```

3.14.8 lightminintensity

Syntaxis:

- [const int] sysinfo.lightminintensity

Descripción

- Propiedad de solo lectura que devuelve una constante que determina el brillo mínimo soportado por el terminal.

3.14.9 lightmaxintensity

Syntaxis:

- [const int] sysinfo.lightmaxintensity

Descripción

- Propiedad de solo lectura que devuelve una constante que determina el brillo máximo soportado por el terminal.

3.14.10 lightdefaultintensity

Syntaxis:

- [const int] sysinfo.lightdefaultintensity

ISBN: 978-84-9012-861-9
Colección Aquilafuente: AQ-241
Ediciones Universidad de Salamanca - cc by nc dc

Descripción

- Propiedad de solo lectura que devuelve una constante que determina el brillo por defecto.

3.14.11 lighttargetprimarydisplayandkeyboard

Syntaxis:

- [const int] sysinfo.lighttargetprimarydisplayandkeyboard
-

Descripción

- Constante para especificar el display primario y la luz del teclado.

3.14.12 lighttargetsystem

Syntaxis:

- [const int] sysinfo.lighttargetsystem

Descripción

- Constante para especificar todos los displays (primario y secundario) y la luz del teclado.

3.14.13 lighton()

Syntaxis:

- [void] sysinfo.lighton(Int lighttarget, Int duration, Int intensity, Bool fadein)

Descripción

- Este método conmuta la luz especificada durante un tiempo e intensidad determinada.

Parámetros de entrada:

- lighttarget: tipo de luz.
- duration: Periodo (en ms.) durante el cual debe encenderse la luz. Si la duración se especifica como *lightinfiniteduration*, la luz permanecerá encendida indefinidamente.
- intensity: Define la intensidad (brillo) de la luz. El valor especificado debe estar

ISBN: 978-84-9012-861-9

Colección Aquilafuente: AQ-241

dentro de los rangos definidos por *lightminintensity* y *lightmaxintensity*.
- fadein: Si el valor es true, enciende la luz utilizando el efecto de fade-in.

Valor de retorno:

- Ninguno

ISBN: 978-84-9012-861-9
Colección Aquilafuente: AQ-241
Ediciones Universidad de Salamanca - cc by nc dc

Ejemplo:

```
functioncallLightOn()
{
var sysinfo = document.embeds[0];
var target = sysinfo.lighttargetsystem; var duration
= sysinfo.lightmaxduration;
var intensity = sysinfo.lightdefaultintensity;
sysinfo.lighton(target, duration, intensity, true);
}
```

ISBN: 978-84-9012-861-9

Colección Aquilafuente: AQ-241

3.14.14 lightoff()

Syntaxis:

- [void] sysinfo.lightoff(String lighttarget, Int duration, Int fadeout)

Descripción

- Este método apaga la luz durante el tiempo especificado.

Parámetros de entrada:

- lighttarget: Tipo de luz a apagar.
- duration: Periodo (en ms.) durante el cual se mantendrá la luz apagada. Si la duración se especifica como lightinfiniteduration, la luz permanecerá apagada indefinidamente.
- fadeout: Si el valor es true, se apagará la luz utilizando el efecto de fade-out. Si es
  false, se apagarán de forma inmediata

Valor de retorno:

- Ninguno

Ejemplo:

```
function callLightOff()
{
var sysinfo = document.embeds[0];
var target = sysinfo.lighttargetsystem; var duration
= 5000; // 5 seconds sysinfo.lightoff(target,dura-
tion,true);
}
```

3.14.15 lightblink()

Syntaxis:

- [void] sysinfo.lightblink(String lighttarget, Int duration, Int onduration, Int offduration, Int intensity)

ISBN: 978-84-9012-861-9
Colección Aquilafuente: AQ-241
Ediciones Universidad de Salamanca - cc by nc dc

Descripción

- Este método enciende y apaga la luz de forma intermitente durante la duración especificada.

Parámetros de entrada:

- lighttarget: Tipo de luz a utilizar.
- duration: Duración (en ms.) durante el cual se mantendrá la luz de forma intermitente. Si la duración se especifica como lightinfiniteduration, la duración de la intermitencia será indefinida.
- onduration: En cada ciclo de intermitencia, la luz estará encendida durante el tiempo especificado por este valor (en ms.)
- offduration: En cada ciclo de intermitencia, la luz estará apagada durante el tiempo
  especificado por este valor (en ms.)
- Define la intensidad (brillo) de la luz. El valor especificado debe estar dentro de los rangos definidos por lightminintensity y lightmaxintensity.

Valor de retorno:

- Ninguno

ISBN: 978-84-9012-861-9

Colección Aquilafuente: AQ-241

Ejemplo:

```
function callLightBlink()
{
// get the Embed element reference var sys-
info = document.embeds[0];
var target = sysinfo.lighttargetsystem; var duration
= 5000; // 5 segundos
var onDur = sysinfo. lightdefaultcycletime; var offDur =
sysinfo. lightdefaultcycletime; var intensity = sys-
info.lightdefaultintensity;
sysinfo.lightblink(target, duration, onDur, offDur, intensity);
}
```


ISBN: 978-84-9012-861-9
Colección Aquilafuente: AQ-241
Ediciones Universidad de Salamanca - cc by nc dc

3.14.16 vibraminintensity

Syntaxis:

- [int] sysinfo.vibraminintensity

Descripción

- Propiedad de sólo-lectura que representa la intensidad mínima de vibración.

3.14.17 vibramaxintensity

Syntaxis:

- [int] sysinfo.vibramaxintensity

Descripción

- Propiedad de sólo-lectura que representa la intensidad máxima de vibración.

3.14.18 vibramaxduration

Syntaxis:

- [int] sysinfo.vibramaxduration

Descripción

- Propiedad de sólo-lectura que representa la duración máxima permitida de vibración.

3.14.19 vibrasettings

Syntaxis:

- [int] sysinfo.vibrasettings

Descripción

- Propiedad de sólo-lectura que devuelve el modo de vibración establecido en el perfil de usuario activo.
- Los valores que puede devolver esta propiedad son los siguientes:
  - 0: Error o no inicializado.
  - 1: Vibrador activado

ISBN: 978-84-9012-861-9

Colección Aquilafuente: AQ-241
Ediciones Universidad de Salamanca - cc by nc dc

- 2: Vibrador desactivado

3.14.20 startvibra()

Syntaxis:

- [void] sysinfo.startvibra(Int duration, Int intensity)

Descripción

- Enciende el vibrador durante el tiempo e intensidad especificados.

Parámetros de entrada:

- duration: Duración en milisegundos. El valor 0 indica duración indefinida.
- intensity: Intensidad de la vibración. Valor debe estar contenido en el rango de -100 a 100. Si el valor es negativo, el motor del vibrador rota en sentido negativo, si no en sentido positivo.

Valor de retorno:

- Ninguno

ISBN: 978-84-9012-861-9
Colección Aquilafuente: AQ-241
Ediciones Universidad de Salamanca - cc by nc dc

Ejemplo:

```
function startVibration()
{
var duration = sysinfo.vibramaxduration; var inten-
sity = sysinfo.vibraminintensity
```

ISBN: 978-84-9012-861-9

Colección Aquilafuente: AQ-241

```
sysinfo.startvibra(duration,  intensity);
}
```

3.14.21 stopvibra()

Syntaxis:

- [void] sysinfo.stopvibra()

Descripción

- Apaga el vibrador de forma inmediata

Parámetros de entrada:

- Ninguno

Valor de retorno:

- Ninguno

ISBN: 978-84-9012-861-9
Colección Aquilafuente: AQ-241
Ediciones Universidad de Salamanca - cc by nc dc

Ejemplo:

```
function callVibration()
{
var duration = 0;
var intensity = sysinfo.vibraminintensity sysinfo.startvi-
bra(duration,  intensity);
// Pulsar cualquier tecla para parar el vibrador
document.addEventListener("keypress", kpListener, false);
}

functionkpListiner(event)
{
sysinfo.stopvibra(); document.removeEventListener("key-
press",kpListener, false);
}
```

ISBN: 978-84-9012-861-9

Colección Aquilafuente: AQ-241

3.14.22 beep()

Syntaxis:

- [void] sysinfo.beep(Int frequency, Int duration)

Descripción

- Reproduce un tono con la frecuencia y duración especificada.

Parámetros de entrada:

- frequency: Frecuencia del tono medida en Hz.
- duration: Duración en ms. Si el valor es 0, entonces el tono durará indefinidamente. o Para parar la reproducción de un tono con duración indefinida, será necesario volver llamar a este método especificando un valor distinto de 0
  para el parámetro duration.

Valor de retorno:

- Ninguno

ISBN: 978-84-9012-861-9
Colección Aquilafuente: AQ-241
Ediciones Universidad de Salamanca - cc by nc dc

Ejemplo:

```
function beepAlert()
{
sysinfo.beep(220, 2000);
}
```

ISBN: 978-84-9012-861-9

Colección Aquilafuente: AQ-241

3.14.23 totalram

Syntaxis:

- [int] sysinfo.totalram

Descripción

- Propiedad de sólo-lectura que devuelve la cantidad total de RAM del terminal, medida en bytes.

Ejemplo:

```
var totalRamSize = sysinfo.totalram;
alert("RAM Total: " + totalRamSize/1024 + "kB");
```

3.14.24 freeram

Syntaxis:

- [int] sysinfo.freeram

Descripción

- Propiedad de sólo-lectura que devuelve la cantidad de RAM libre del terminal, medida en bytes.

Ejemplo:

```
var freeram= sysinfo.freeram;
alert("RAM Libre: " + freeram/1024 + "kB");
```

3.14.25 drivelist

Syntaxis:

- [string] sysinfo.drivelist

Descripción

- Propiedad de sólo-lectura que devuelve los nombres de las unidades de usuario separadas por un espacio.

ISBN: 978-84-9012-861-9
Colección Aquilafuente: AQ-241
Ediciones Universidad de Salamanca - cc by nc dc

- Las unidades z: y d: son unidades del sistema y por lo tanto no son devueltas por esta propiedad.

3.14.26 drivesize()

Syntaxis:

- [int] sysinfo.drivesize(String drivename)

Descripción

- Este método permite conocer el tamaño total de la unidad especificada.

Parámetros de entrada:

- drivename: Unidad a ser examinada

Valor de retorno:

- Tamaño total (en bytes) de la unidad.

3.14.27 drivefree ()

Syntaxis:

- [int] sysinfo.drivefree(String drivename)

Descripción

ISBN: 978-84-9012-861-9

Colección Aquilafuente: AQ-241

- Este método permite conocer el espacio libre que hay disponible en la unidad especificada.

Parámetros de entrada:

- drivename: Unidad a ser examinada

Valor de retorno:

- Espacio libre (en bytes) de la unidad.

Ejemplo:

```
function checkDrivesInformation() { var space = 0;
var allDrives = sysinfo.drivelist; var drives = allDrives.split("
");

 for (var i=0; i<drives.length; i++) {
        space=sysinfo.drivesize(drives[i]); space /=1024; // bytes ->
        kB
        alert("Total"+drives[i]+"="+space+"kB"; space=sysinfo.drive-
        free(drives[i]); space /=1024; // bytes -> kB
        alert("Libre"+drives[i]+"="+space+"kB";
                }
               }
```

3.14.28 language

Syntaxis:

- [string]sysinfo.language

Descripción

- Propiedad de sólo lectura que retorna el idioma utilizado por el terminal en formato ISO 639-1. Por ejemplo:
  - Inglés: en
  - Alemán: de
  - Español: es

ISBN: 978-84-9012-861-9
Colección Aquilafuente: AQ-241
Ediciones Universidad de Salamanca - cc by nc dc

References

1. António Pereira, Filipe Felisberto, Luis Maduro, Miguel Felgueiras (2012). Fall Detection on Ambient Assisted Living using a Wireless Sensor Network. ADCAIJ: Advances in Distributed Computing and Artificial Intelligence Journal (ISSN: 2255-2863), Salamanca, v. 1, n. 1
2. Baruque, B., Corchado, E., Mata, A., & Corchado, J. M. (2010). A forecasting solution to the oil spill problem based on a hybrid intelligent system. Information Sciences, 180(10), 2029–2043. https://doi.org/10.1016/j.ins.2009.12.032
3. Canizes, B., Pinto, T., Soares, J., Vale, Z., Chamoso, P., & Santos, D. (2017). Smart City: A GECAD-BISITE Energy Management Case Study. In 15th International Conference on Practical Applications of Agents and Multi-Agent Systems PAAMS 2017, Trends in Cyber-Physical Multi-Agent Systems (Vol. 2, pp. 92–100). https://doi.org/10.1007/978-3-319-61578-3_9
4. Casado-Vara, R., Chamoso, P., De la Prieta, F., Prieto J., & Corchado J.M. (2019). Non-linear adaptive closed-loop control system for improved efficiency in IoT-blockchain management. Information Fusion.
5. Casado-Vara, R., Novais, P., Gil, A. B., Prieto, J., & Corchado, J. M. (2019). Distributed continuous-time fault estimation control for multiple devices in IoT networks. IEEE Access.
6. Casado-Vara, R., Vale, Z., Prieto, J., & Corchado, J. (2018). Fault-tolerant temperature control algorithm for IoT networks in smart buildings. Energies, 11(12), 3430.
7. Casado-Vara, R., Prieto-Castrillo, F., & Corchado, J. M. (2018). A game theory approach for cooperative control to improve data quality and false data detection in WSN. International Journal of Robust and Nonlinear Control, 28(16), 5087-5102.
8. Chamoso, P., de La Prieta, F., Eibenstein, A., Santos-Santos, D., Tizio, A., & Vittorini, P. (2017). A device supporting the self-management of tinnitus. In Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics) (Vol. 10209 LNCS, pp. 399–410). https://doi.org/10.1007/978-3-319-56154-7_36
9. Chamoso, P., González-Briones, A., Rivas, A., De La Prieta, F., & Corchado J.M. (2019). Social computing in currency exchange. Knowledge and Information Systems.
10. Chamoso, P., González-Briones, A., Rivas, A., De La Prieta, F., & Corchado, J. M. (2019). Social computing in currency exchange. Knowledge and Information Systems, 1-21.
11. Chamoso, P., González-Briones, A., Rodríguez, S., & Corchado, J. M. (2018). Tendencies of technologies and platforms in smart cities: A state-of-the-art review. Wireless Communications and Mobile Computing, 2018.
12. Chamoso, P., Rodríguez, S., de la Prieta, F., & Bajo, J. (2018). Classification of retinal vessels using a collaborative agent-based architecture. AI Communications, (Preprint), 1-18.
13. Choon, Y. W., Mohamad, M. S., Deris, S., Illias, R. M., Chong, C. K., Chai, L. E., … Corchado, J. M. (2014). Differential bees flux balance analysis with OptKnock for in silico microbial strains optimization. PLoS ONE, 9(7). https://doi.org/10.1371/journal.pone.0102744
14. Corchado, J. M., & Aiken, J. (2002). Hybrid artificial intelligence methods in oceanographic forecast models. Ieee Transactions on Systems Man and Cybernetics Part C-Applications and Reviews, 32(4), 307–313. https://doi.org/10.1109/tsmcc.2002.806072
15. David Griol, Jesús García-Herrero, José Manuel Molina (2013). Combining heterogeneous inputs for the development of adaptive and multimodal interaction systems. ADCAIJ: Advances in Distributed Computing and Artificial Intelligence Journal (ISSN: 2255-2863), Salamanca, v. 2, n. 3
16. Fdez-Riverola, F., & Corchado, J. M. (2003). CBR based system for forecasting red tides. Knowledge-Based Systems, 16(5–6 SPEC.), 321–328. https://doi.org/10.1016/S0950-7051(03)00034-0
17. Fernández-Riverola, F., Díaz, F., & Corchado, J. M. (2007). Reducing the memory size of a Fuzzy case-based reasoning system applying rough set techniques. IEEE Transactions on Systems, Man and Cybernetics Part C: Applications and Reviews, 37(1), 138–146. https://doi.org/10.1109/TSMCC.2006.876058
18. Fyfe, C., & Corchado, J. (2002). A comparison of Kernel methods for instantiating case based reasoning systems. Advanced Engineering Informatics, 16(3), 165–178. https://doi.org/10.1016/S1474-0346(02)00008-3
19. Fyfe, C., & Corchado, J. M. (2001). Automating the construction of CBR systems using kernel methods. International Journal of Intelligent Systems, 16(4), 571–586. https://doi.org/10.1002/int.1024

ISBN: 978-84-9012-861-9

Colección Aquilafuente: AQ-241
Ediciones Universidad de Salamanca - cc by nc dc

20. García Coria, J. A., Castellanos-Garzón, J. A., & Corchado, J. M. (2014). Intelligent business processes composition based on multi-agent systems. Expert Systems with Applications, 41(4 PART 1), 1189–1205. https://doi.org/10.1016/j.eswa.2013.08.003
21. García, O., Chamoso, P., Prieto, J., Rodríguez, S., & De La Prieta, F. (2017). A serious game to reduce consumption in smart buildings. In Communications in Computer and Information Science (Vol. 722, pp. 481–493). https://doi.org/10.1007/978-3-319-60285-1_41
22. Glez-Bedia, M., Corchado, J. M., Corchado, E. S., & Fyfe, C. (2002). Analytical model for constructing deliberative agents. International Journal of Engineering Intelligent Systems for Electrical Engineering and Communications, 10(3).
23. Glez-Peña, D., Díaz, F., Hernández, J. M., Corchado, J. M., & Fdez-Riverola, F. (2009). geneCBR: A translational tool for multiple-microarray analysis and integrative information retrieval for aiding diagnosis in cancer research. BMC Bioinformatics, 10. https://doi.org/10.1186/1471-2105-10-187
24. Gonzalez-Briones, A., Chamoso, P., De La Prieta, F., Demazeau, Y., & Corchado, J. M. (2018). Agreement Technologies for Energy Optimization at Home. Sensors (Basel), 18(5), 1633-1633. doi:10.3390/s18051633
25. González-Briones, A., Chamoso, P., Yoe, H., & Corchado, J. M. (2018). GreenVMAS: virtual organization-based platform for heating greenhouses using waste energy from power plants. Sensors, 18(3), 861.
26. Gonzalez-Briones, A., Prieto, J., De La Prieta, F., Herrera-Viedma, E., & Corchado, J. M. (2018). Energy Optimization Using a Case-Based Reasoning Strategy. Sensors (Basel), 18(3), 865-865. doi:10.3390/s18030865
27. Li, T., Sun, S., Corchado, J. M., & Siyau, M. F. (2014). A particle dyeing approach for track continuity for the SMC-PHD filter. In FUSION 2014 - 17th International Conference on Information Fusion. Retrieved from https://www.scopus.com/inward/record.uri?eid=2-s2.0-84910637583&partnerID=40&md5=709eb4815eaf544ce01a2c21aa749d8f
28. Lima, A. C. E. S., De Castro, L. N., & Corchado, J. M. (2015). A polarity analysis framework for Twitter messages. Applied Mathematics and Computation, 270, 756–767. https://doi.org/10.1016/j.amc.2015.08.059
29. Mata, A., & Corchado, J. M. (2009). Forecasting the probability of finding oil slicks using a CBR system. Expert Systems with Applications, 36(4), 8239–8246. https://doi.org/10.1016/j.eswa.2008.10.003
30. Méndez, J. R., Fdez-Riverola, F., Díaz, F., Iglesias, E. L., & Corchado, J. M. (2006). A comparative performance study of feature selection methods for the anti-spam filtering domain. Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), 4065 LNAI, 106–120. Retrieved from https://www.scopus.com/inward/record.uri?eid=2-s2.0-33746435792&partnerID=40&md5=25345ac884f61c182680241828d448c5
31. Pablo Campillo-Sánchez, Juan Antonio Botía, Jorge Gómez-Sanza (2013). Development of Sensor Based Applications for the Android Platform: an Approach Based on Realistic Simulation. ADCAIJ: Advances in Distributed Computing and Artificial Intelligence Journal (ISSN: 2255-2863), Salamanca, v. 2, n. 1
32. Palomino, C. G., Nunes, C. S., Silveira, R. A., González, S. R., & Nakayama, M. K. (2017). Adaptive agent-based environment model to enable the teacher to create an adaptive class. Advances in Intelligent Systems and Computing (Vol. 617). https://doi.org/10.1007/978-3-319-60819-8_3
33. Sigeru Omatu, Hideo Araki, Toru Fujinaka, Mitsuaki Yano, Michifumi Yoshioka, Hiroyuki Nakazumi, Ichiro Tanahashi (2012). Mixed Odor Classification for QCM Sensor Data by Neural Network. ADCAIJ: Advances in Distributed Computing and Artificial Intelligence Journal (ISSN: 2255-2863), Salamanca, v. 1, n. 2
34. Tapia, D. I., & Corchado, J. M. (2009). An ambient intelligence based multi-agent system for alzheimer health care. International Journal of Ambient Computing and Intelligence, v 1, n 1(1), 15–26. https://doi.org/10.4018/jaci.2009010102
35. Xian Wang, Paula Tarrío, Ana María Bernardos, Eduardo Metola, José Ramón Casar (2012). User-independent accelerometer-based gesture recognition for mobile devices. ADCAIJ: Advances in Distributed Computing and Artificial Intelligence Journal (ISSN: 2255-2863), Salamanca, v. 1, n. 3
36. Angelo Costa, Stella Heras, Javier Palanca, Paulo Novais, Vicente Julián (2016). Persuasion and Recommendation System Applied to a Cognitive Assistant. ADCAIJ: Advances in Distributed Computing and Artificial Intelligence Journal (ISSN: 2255-2863), Salamanca, v. 5, n. 2
37. David Griol, Jose M. Molina (2016). A proposal to manage multi-task dialogs in conversational interfaces. ADCAIJ: Advances in Distributed Computing and Artificial Intelligence Journal (ISSN: 2255-2863), Salamanca, v. 5, n. 2
38. Marco Antonio Ameller, María Angélica González (2016). Minutiae filtering using ridge-valley method. ADCAIJ: Advances in Distributed Computing and Artificial Intelligence Journal (ISSN: 2255-2863), Salamanca, v. 5, n. 1

ISBN: 978-84-9012-861-9
Colección Aquilafuente: AQ-241
Ediciones Universidad de Salamanca - cc by nc dc

39. Elton S Siqueira, Patrick Cisuaka Kabongo, Tiancheng Li, Carla D. Castanho, Li Weigang (2016). On Chinese and Western Family Trees: Mechanism and Performance. ADCAIJ: Advances in Distributed Computing and Artificial Intelligence Journal (ISSN: 2255-2863), Salamanca, v. 5, n. 1
40. Eduardo Facchini, Eduardo Mario Dias, Alexandre Pelegi Abreu, Maria Lídia Rebello Pinho Dias (2016). Brazil in Search of Transparency E-Gov. ADCAIJ: Advances in Distributed Computing and Artificial Intelligence Journal (ISSN: 2255-2863), Salamanca, v. 5, n. 1
41. Ana Oliveira Alves, Tiago Dias, David Silva (2015). A Real-Time, Distributed and Context-Aware System for Managing Solidarity Campaigns. ADCAIJ: Advances in Distributed Computing and Artificial Intelligence Journal (ISSN: 2255-2863), Salamanca, v. 4, n. 2
42. Eduardo Mario Dias, Eduardo Facchini, Antônio Carlos De Moraes, Mauricio Lima Ferreira, Willian Reginato Este, Maria Lídia Rebello, Pinho Dias (2014). A Future Look. ADCAIJ: Advances in Distributed Computing and Artificial Intelligence Journal (ISSN: 2255-2863), Salamanca, v. 3, n. 3
43. Carlos Alberto Ochoa, Lourdes Yolanda Margain, Francisco Javier Ornelas, Sandra Guadalupe Jiménez, Teresa Guadalupe Padilla (2014). Using multi-objective optimization to design parameters in electro-discharge machining by wire. ADCAIJ: Advances in Distributed Computing and Artificial Intelligence Journal (ISSN: 2255-2863), Salamanca, v. 3, n. 2
44. Vanessa N. Cooper, Hisham M. Haddad, Hossain Shahriar (2014). Android Malware Detection Using Kullback-Leibler Divergence. ADCAIJ: Advances in Distributed Computing and Artificial Intelligence Journal (ISSN: 2255-2863), Salamanca, v. 3, n. 2
45. Saverio Giallorenzo, Maurizio Gabbrielli, Fabrizio Montesi (2014). Service-Oriented Architectures: from Design to Production exploiting Workflow Patterns. ADCAIJ: Advances in Distributed Computing and Artificial Intelligence Journal (ISSN: 2255-2863), Salamanca, v. 3, n. 2
46. Muhammad Amin Khan, Felix Freitag (2014). Sparks in the Fog: Social and Economic Mechanisms as Enablers for Community Network Clouds. ADCAIJ: Advances in Distributed Computing and Artificial Intelligence Journal (ISSN: 2255-2863), Salamanca, v. 3, n. 1
47. Johannes Fähndrich, Sebastian Ahrndt, Sahin Albayrak (2014). Formal Language Decomposition into Semantic Primes. ADCAIJ: Advances in Distributed Computing and Artificial Intelligence Journal (ISSN: 2255-2863), Salamanca, v. 3, n. 1
48. Merce Teixido, Tomás Palleja, Marcel Tresanchez, Davinia Font, Javier Moreno, Alicia Fernández, Jordi Palacín, Carlos Rebate (2013). Optimization of the virtual mouse HeadMouse to foster its classroom use by children with physical disabilities. ADCAIJ: Advances in Distributed Computing and Artificial Intelligence Journal (ISSN: 2255-2863), Salamanca, v. 2, n. 4
49. Ana Karin Chávez Valdivia (2017). Between the Profiles Pay Per View and the Protection of Personal Data: the Product is You. ADCAIJ: Advances in Distributed Computing and Artificial Intelligence Journal (ISSN: 2255-2863), Salamanca, v. 6, n. 1
50. Pawel Pawlewski, Kamila Kluska (2017). Modeling and simulation of bus assembling process using DES/ABS approach. ADCAIJ: Advances in Distributed Computing and Artificial Intelligence Journal (ISSN: 2255-2863), Salamanca, v. 6, n. 1
51. Davide Carneiro, Daniel Araújo, André Pimenta, Paulo Novais (2016). Real Time Analytics for Characterizing the Computer User's State. ADCAIJ: Advances in Distributed Computing and Artificial Intelligence Journal (ISSN: 2255-2863), Salamanca, v. 5, n. 4
52. Roussanka Loukanova (2016). Relationships between Specified and Underspecified Quantification by the Theory of Acyclic Recursion. ADCAIJ: Advances in Distributed Computing and Artificial Intelligence Journal (ISSN: 2255-2863), Salamanca, v. 5, n. 4
53. David Griol, Jose Manuel Molina (2016). From VoiceXML to multimodal mobile Apps: development of practical conversational interfaces. ADCAIJ: Advances in Distributed Computing and Artificial Intelligence Journal (ISSN: 2255-2863), Salamanca, v. 5, n. 3

ISBN: 978-84-9012-861-9

Colección Aquilafuente: AQ-241
Ediciones Universidad de Salamanca - cc by nc dc