EUSKO JAURLARITZA
GOBIERNO VASCO

Programa Saiotek 2006

SMARTLAB

Entorno de Trabajo Inteligente Colaborativo y Programable

Manual del Programador de Gadgets correspondientes a Servicios Avanzados en SmartLab

Tecnológico Fundación Deusto
Teknologikoa Deustu Fundazioa

HISTORIAL DE CAMBIOS

Versión	Descripción	Autor	Fecha	Comentarios
V0.1	Versión inicial	Ander Barbier	23/01/2008	

TABLA DE CONTENIDOS

1 INTRODUCCIÓN

Para la creación de la interfaz de usuario se ha desarrollado una arquitectura de gadgets, donde la ventana principal cuenta con una zona contenedora de pequeñas ventanas o gadgets, que presentan la interfaz de usuario particular de cada dispositivo o elemento conectado al sistema. Este sistema basado en gadgets es totalmente modular, facilitando la creación e incorporación de nuevas interfaces de usuario para el control y/o monitorización de nuevos dispositivos conectados al sistema.

La interfaz de usuario del proyecto SmartLab, está desarrollada con tecnologías web, siguiendo los estándares del World Wide Web Consortium XHTML-1.0-Strict [http://www.w3.org/TR/xhtml1/#a_dtd_XHTML-1.0-Strict], ,CSS2 [http://www.w3.org/TR/CSS21/] y Javascript como lenguaje de scripting. Por lo tanto, cualquier navegador que respete los estándares del W3C será el único requisito para poder utilizar el sistema.

La creación de un nuevo gadget es sencilla, y únicamente requiere conocimientos de las mencionadas tecnologías de diseño web como xhtml, css y javascript. A continuación se presenta una pequeña guía para los creadores de nuevos gadgets.

1.1 Contenido de un gadget

Cada gadget se carga dentro de un Iframe dentro de la página principal por lo que un documento html totalmente independiente contiene el contendido de cada uno de los gadgtes. Para visualizar su contenido, el sistema carga el documento html del gadget en el iframe correspondiente de la ventana principal.

La interfaz de usuario de cada gadget además del contenido, necesita otros elementos como los estilos correspondientes para dar formato a los contenidos, o el mecanismo de gestión de eventos que permite interactuar al usuario con el sistema, etc. Por tanto, cada gadget estará formado por un conjunto de ficheros:

1.1.1 Documento html

Recoge el contenido del gadget. Este documento se tiene que llamar index.htm. El código de ejemplo que se muestra a continación muestra el contenido del documento html de uno

de los gadgets más sencillos, el gadget Thermometer que simplemente visualiza la temperatura de un termómetro:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en" lang="en">
<head>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1" />
<title>SmartLab - Tecológico Fundación Deusto</title>
<link rel='stylesheet' type='text/css'
href='/resources/styles/SmartLab/SmartLab_gadgets.css'/>
<link rel='stylesheet' type='text/css' href='<%baseUrl%>/thermometer.css'/>
<script type='text/javascript' src='/resources/scripts/crossbrowser/x_core.js'></script>
<script type='text/javascript' src='/resources/scripts/crossbrowser/x_event.js'></script>
<script type='text/javascript' src='/resources/scripts/crossbrowser/xenabledrag.js'></script>
<script type='text/javascript'
src='/resources/scripts/crossbrowser/xgetelementbyid.js'></script>
<script type='text/javascript'
src='/resources/scripts/crossbrowser/xgetelementsbytagname.js'></script>
<script type='text/javascript' src='/resources/scripts/crossbrowser/xhttprequest.js'></script>
<script type="text/javascript" src="/resources/scripts/SmartLab/gadgets.js"></script>
</head>
<body>
<div class='thermometer-content'>
       <div id="gadgetTitle" class='thermometer-title'> </div>
       <div id="temp-text"> </div>
       <div id='lblError'>
       </div>
</div>
<script type="text/javascript">

var URL="<%servletUrl%>";
var serviceId = "<%serviceId%>";
var title="<%title%>";
var baseUrl="<%baseUrl%>";

</script>

<script type="text/javascript" src="<%baseUrl%>/thermometer.js"></script>
</body>
</html>
```

El HTML obligatoriamente debe contener los siguientes elementos:

- Referencia a una hoja de estilos general del sistema de gadgets:

```
<link rel='stylesheet' type='text/css'
href='/resources/styles/SmartLab/SmartLab_gadgets.css'/>
```

- Referencia al fichero de estilos css del gadget:

```
<link rel='stylesheet' type='text/css' href='<%baseUrl%>/thermometer.css'/>
```

- Referencia a librerías javascript utilizadas por la arquitectura de gadgets de SmartLab:

```
<script type='text/javascript' src='/resources/scripts/crossbrowser/x_core.js'></script>
<script type='text/javascript' src='/resources/scripts/crossbrowser/x_event.js'></script>
<script type='text/javascript' src='/resources/scripts/crossbrowser/xenabledrag.js'></script>
<script type='text/javascript'
src='/resources/scripts/crossbrowser/xgetelementbyid.js'></script>
<script type='text/javascript'
src='/resources/scripts/crossbrowser/xgetelementsbytagname.js'></script>
<script type='text/javascript' src='/resources/scripts/crossbrowser/xhttprequest.js'></script>
<script type="text/javascript"
src="/SmartLab/gadgets/light/debug/scriptaculo/prototype.js"></script>
<script type="text/javascript"
src="/SmartLab/gadgets/light/debug/scriptaculo/scriptaculous.js"></script>
<script type="text/javascript" src="/resources/scripts/SmartLab/gadgets.js"></script>
```

- Contenido del gadget. El contenido se recomienda ponerlo dentro de un div en el cuerpo del documento:

```
<div class='thermometer-content'>
```

```
...
</div>
```

- Script con los parámetros del gadget. Este pequeño script carga unas variables javascript, que contienen la información particular del dispositivo o elemento particular controlado por el gadget. El SmartLabServlet , antes de enivar el html al navegador sustituye los siguientes códigos: <%servletUrl%>, <%serviceId%>, <%serviceId%>, <%title%>, <%baseUrl%>, por los valores correspondientes al dispositivo controlado:

```
<script type="text/javascript">

var URL="<%servletUrl%>";
var serviceId = "<%serviceId%>";
var title="<%title%>";
var baseUrl="<%baseUrl%>";

</script>
```

- Referencia al script del gadget, que se explica más adelante:

```
<script type="text/javascript" src="<%baseUrl%>/thermometer.js"></script>
```

1.1.2 Hoja de estilos CSS

La hoja de estilos contiene el formato de todos los elementos contenidos en el gadget. No existe ningún tipo de requisito en cuanto a los estilos, por lo que el diseñador del gadget puede crear todos los estilos que considere necesarios. Se recomienda crear los estilos en un fichero externo con el nombre del gadget. A modo de ejemplo, se muestra la hoja de estilos del gadget Thermometer (thermometer.css):

```
html{
background-color:#000000;
}
.thermometer-content{
	background-color:#000000;
}
#temp-text{
	color:#fb9b29;
	font-size:3em;
	padding:10px;
	text-align:center;
}
.thermometer-title{
	color:#fb9b29;
	font-size:1.5em;
	padding:10px;
	text-align:center;
}
.thermometer-error
	{
	text-align:center;
	font-size:1em;
	color:#aaa;
	}
```

1.1.3 Script del gadget

Todos los gadgets requieren un script formado por varias funciones que se encarga de recoger y gestionar los eventos realizados por el usuario. Este script necesita conectarse

con el servidor del sistema para enviar las peticiones del usuario y para leer periódicamente el estado del dispositivo. Se recomienda crear este script en un fichero externo con el nombre del gadget. A continuación se muestra el script completo del gadget Thermometer (thermoter.js):

```
getTempCall=null;
loadTitle();
getTempValue();

function loadTitle()
{
	document.getElementById("gadgetTitle").innerHTML=title;
}

function setTemp(temp)
{

		document.getElementById("temp-
text").innerHTML=Math.round(parseFloat(temp)*10)/10 + " ºC";
}

function showError(message)
{
	lblError.innerHTML="<p class='thermometer-error'>" + message + "</p>";
}

function getTempValue()
{
	var method;
	var params;
	method='getTemp';
	params = [];
	getTempCall=new Invocacion(URL, serviceId, method, params, getTempValueResponse,null)
	getTempCall.enviar();
	setTimeout("getTempValue()", 5000);
}

function getTempValueResponse(req, status, thermometer_Object)
{
	var temperature_Temp;
	if (status == getTempCall.XHR.OK)
	{
		if(req.responseXML.getElementsByTagName("value").length==1)
			{

	thermometer_Temp=req.responseXML.getElementsByTagName("value")[0].childNodes[0].data;
			setTemp(thermometer_Temp);
		}
  }
  else
	{
    if (status & getTempCall.XHR.TIMEOUT)
     showError("Timeout error");

    if (status & getTempCall.XHR.NOXMLCT)
     showError("Noxmlct error");

    if (status & getTempCall.XHR.RSPERR)
      showError();
    }
}
```

Lo primero que tienen que hacer todos los gadgets es visualizar el título a partir de la variable title, cargada en el script de parámeteros ubicado en documento html del gadget. Se recomienda llamar a este método loadTitle. Ejemplo del loadTitle del gadget Thermometer:

```
function loadTitle()
{
	document.getElementById("gadgetTitle").innerHTML=title;
}
```

Otra operación que tienen que hacer todos los gadgets es leer el estado del dispositivo periódicamente. El estado de los gadgets se pueden obtener invocando métodos a la plataforma SmartLab. Los métodos se invocan a través del Servlet, que es el encargado de la comunicación entre la interfaz Web y la plataforma SmartLab. La invocación de métodos se realiza mediante peticiones http al mencionando Servlet. Toda la lógica de las peticiones AJAX se encuentra encapsulada en un objeto Javascript llamado Invocacion, que se encuentra en un .js ya referenciado en el html del gadget (/resources/scripts/SmartLab/gadgets.js). Este objeto invocación tiene un contructor con la siguiente sintaxis:

```
new Invocacion(URL, serviceId, method, params, fnCallback, UData)
```

Parámetros:

- URL. String. URL a la que se va a hacer la petición.
- serviceId. Identificativo del dispositivo controlado por el gadget.
- method. String. Método de la plataforma SmartLab que se quiere a invocar.
- params. Array. Parámetros que se van a pasar al método invocado.
- fnCallback. Función javascript que se va a llamar cuando se reciba la respuesta o cuando expire el timeout. Esta función recibe tres parámetros fnCallback(req, status, data);
    - req – El objeto XMLHttpRequest.
    - status – El estado de la petición http.
    - data – el objeto de usuario pasado.
- UData. El objeto de usuario que se pasará a la función de retorno (fnCallback).

El objeto Invocacion también tiene un método sin parámetros llamado enviar que realiza la petición http:

```
enviar()
```

A continuación se muestra como el gadget Thermometer recoge la temperatura periódicamente haciendo uso del mencionado objeto Invocacion:

```
function getTempValue()
```

```
{
	var method;
	var params;
	method='getTemp';
	params = [];
	getTempCall=new Invocacion(URL, serviceId, method, params, getTempValueResponse,null)
	getTempCall.enviar();
	setTimeout("getTempValue()", 5000);
}
```

La función getTempValue lee la temperatura del dispositivo. Como se tiene que leer la temperatura periódicamente, la última instrucción de la función es una llamada a si misma a través de la función setTimeOut de javascript.

En la llamada al constructor del objeto Invocacion, la URL y el serviceId están almacenados en las variables cargadas en el script de parámetros del documento html. El método que hay que invocar en la plataforma Smartalb para obtener la temperatura se llama getTemp. En este caso el método no recibe ningún parámetro por lo que el Array se pasa vacío, pero si el método que se quiere invocar necesita parámetros el argumento params debe ser un Array bidimensional con el tipo y valor de cada uno de los parámetros. Ejemplo:

```
Params=[ ['Integer', 1], ['String', 'on' ]];
```

La función de retorno se llama getTempValueResponse, que será llamada cuando se reciba la respuesta o cuando expire el timeout. Si en la función de retorno fuera necesario referenciar a un elemento del documento html se podría pasar la referencia al elemento a través del parámetro UData. En el método getTempValue no es necesario por lo que se pasa a null.

A continuación se muestra el código de la función de reterno getTempValueResponse que se encarga de recoger la petición:

```
function getTempValueResponse(req, status, thermometer_Object)
{
	var temperature_Temp;
	if (status == getTempCall.XHR.OK)
	{
		if(req.responseXML.getElementsByTagName("value").length==1)
			{

	thermometer_Temp=req.responseXML.getElementsByTagName("value")[0].childNodes[0].data;
			setTemp(thermometer_Temp);
		}
  }
  else
	{
    if (status & getTempCall.XHR.TIMEOUT)
     showError("Timeout error");

    if (status & getTempCall.XHR.NOXMLCT)
     showError("Noxmlct error");

    if (status & getTempCall.XHR.RSPERR)
      showError("Response error");
    }
}
```

Esta función, si el estado de la petición es OK, recoge la temperatura parseando el XML recibido a utilizando XMLDOM. A continuación se llama al método javascript setTemp que lo único que hace es modificar el innerHTML del elemento que visualiza la temperatura.
Si el estado de la petición no es OK, se llama a una función showError que muestra el error correspondiente.

El gadget de ejemplo es muy sencillo, y la información sólo fluye en una dirección: de la plataforma al usuario. Si hubiese sido necesario que el usuario controlase el dispositivo, por ejemplo subiendo o bajando la temperatura, simplemente habría que realizar la Invocación al método correspondiente a través del objeto Invocacion de la misma forma que se hace en el ejemplo mostrado, pero pasando el nombre del método y los parámetros necesarios para que la plataforma SmartLab realice la operación correspondiente sobre el dispositivo.

1.1.4 Otros ficheros.

Si la interfaz de usuario requiere otros recursos como imágenes, u otras librerías javascript que resuelvan una problemática concreta del gadget, simplemente habría que añadir los ficheros correspondientes.