



Taller Desarrollando aplicaciones con Bluevia y Java

Versión 1.0

Junio 30 de 2011

L I M I N A L

TABLA DE CONTENIDO

1. OBJETIVOS	3
2. INTRODUCCIÓN	3
3. REQUISITOS	3
4. PREPARACIÓN DEL AMBIENTE	3
5. REGISTRO EN BLUEVIA	6
ACTIVACIÓN DE CUENTA DE CLIENTE	9
OBTENCIÓN DE API KEYS	9
6. AUTENTICACIÓN Y PERMISOS	12
7. ENVIANDO Y RECIBIENDO MENSAJES	16
ENVÍO DE SMS	16
RECEPCIÓN DE SMS	17
ENVÍO DE MMS	18
RECEPCIÓN DE MMS	19
8. PUBLICIDAD	21
9. EJERCICIO PRÁCTICO	21

1. OBJETIVOS

- Conocer, de modo práctico, el uso del SDK de Java para el desarrollo de aplicaciones utilizando Bluevia.
- Desarrollar una aplicación stand-alone sencilla utilizando el SDK.
- Implementar una aplicación web sencilla utilizando los servicios de Bluevia.

2. INTRODUCCIÓN

Las nuevas necesidades que surgen a través del uso de la tecnología llevan al desarrollo de nuevos modelos e implementaciones. Bluevia es una propuesta nueva para permitir a los desarrolladores desarrollar sus aplicaciones, independientes de lenguaje o plataforma, para que interactúen con los millones de usuario de Movistar-O2 independiente del móvil que posean. En Colombia el servicio ya se presta en modo beta para desarrolladores y se espera que en los próximos meses este totalmente activado para su uso comercial. El presente taller describe el desarrollo de una aplicación sencilla en Java que utiliza los servicios de la plataforma Bluevia habilitados para Colombia, además al final se propone un ejercicio práctico para ser desarrollado.

3. REQUISITOS

- Conocimiento del lenguaje Java para el desarrollo de aplicaciones Stand-Alone y Web.
- Computador portátil con Netbeans 7.0 y servidor Tomcat
- Teléfono móvil con SIM Card de Telefónica y una precarga sugerida de \$10.000
- Acceso a Internet

4. PREPARACIÓN DEL AMBIENTE

Antes de comenzar a trabajar en el desarrollo de aplicaciones que utilicen la plataforma de Bluevia es necesario establecer el ambiente de trabajo. Esta preparación implica los siguientes elementos:

- Descargar e instalar el IDE Netbeans 7.0, lo puede encontrar en la [página web de Netbeans](#). Es recomendable descargar el instalador que contiene Tomcat ya que se utilizará en el taller propuesto.
- Descargar el SDK de Bluevia para Java, se puede [descargar de la página de Bluevia](#).
- Registrarse en la página de Bluevia como se describirá a continuación.
- Registrar el número celular en la página de [Bluevia Connect](#), como se presentará a continuación.

Una vez completadas estas solicitudes se procede a preparar nuestro entorno de desarrollo. Para esto creamos un nuevo proyecto en Netbeans, para este primer ejemplo debe ser una aplicación Java sencilla, como se muestra en la Figura 1.

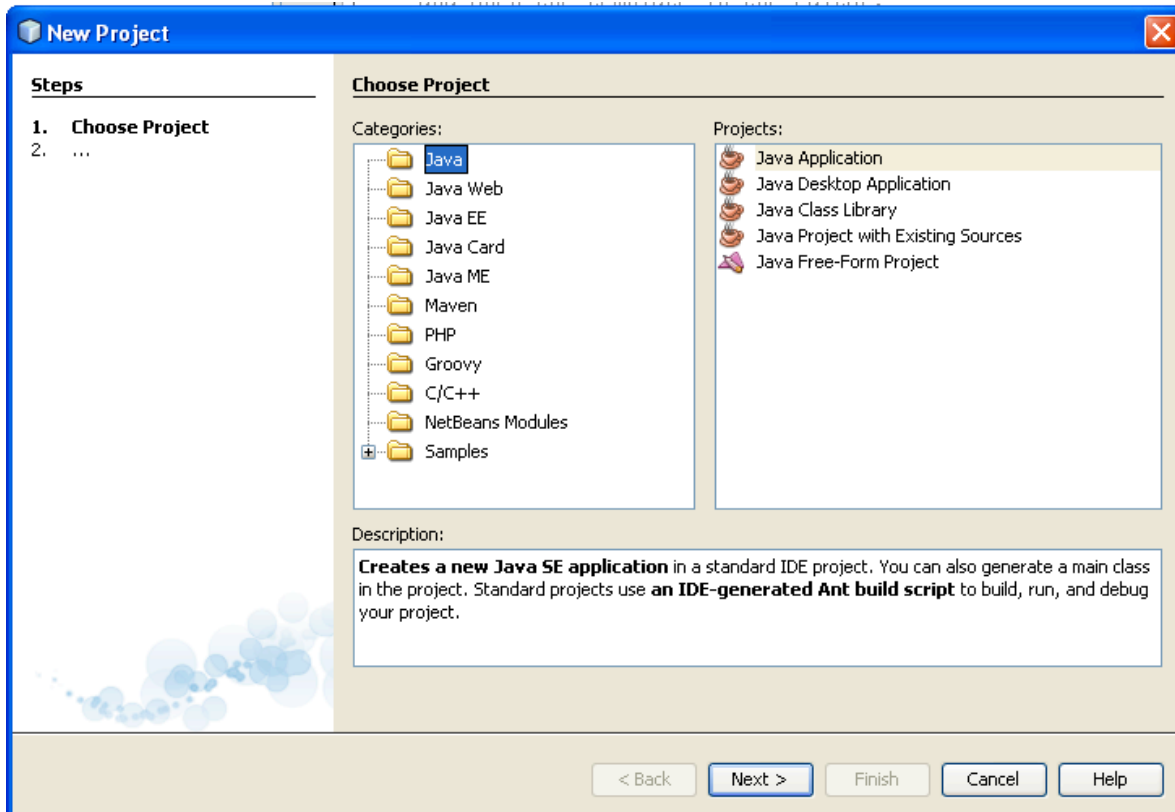
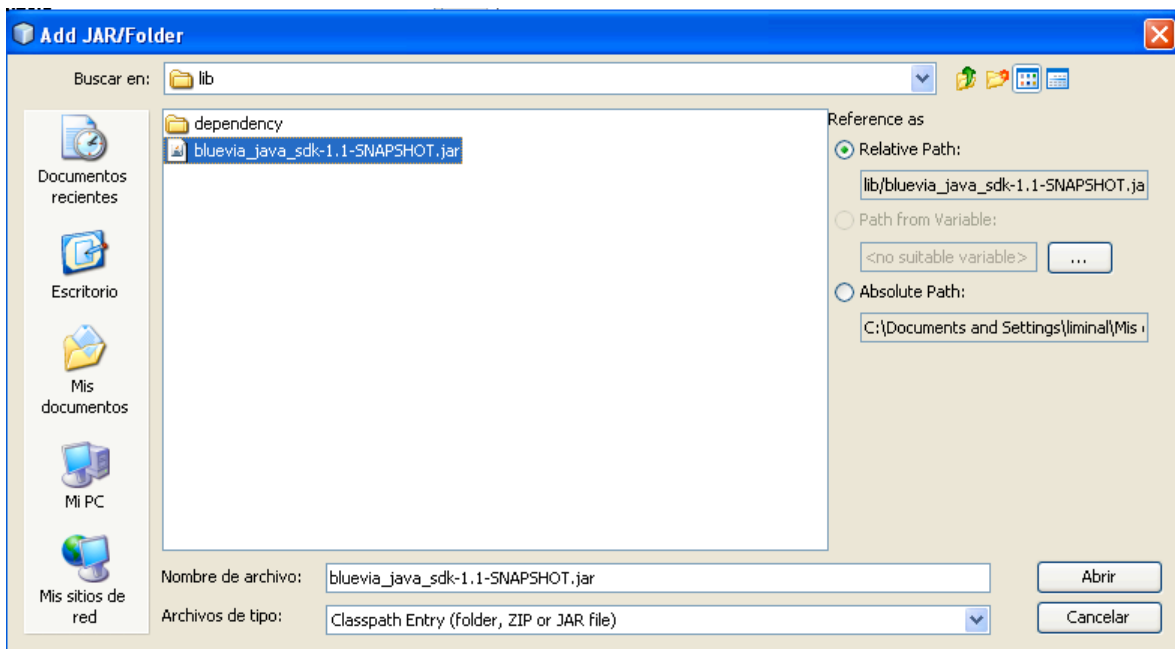


Figura 1 Creación del proyecto en Netbeans

Le asignamos un nombre a nuestro proyecto, que para este caso será "TallerBluevia" y creamos el proyecto. Es importante tener en cuenta donde creamos el proyecto ya que en esta carpeta es en donde debemos agregar los JARs de Bluevia.

Para agregar el SDK primero debemos descomprimir el bluevia-sdk-java-v1.3.zip que descargamos de la [página de Bluevia](#). Esto genera una carpeta con una estructura donde podemos encontrar los JARs, la documentación de estos y algunos ejemplos. Ahora en el proyecto en Netbeans agregamos estos JARs como librerías. Primero debemos crear en la carpeta de nuestro proyecto una carpeta a la que llamaremos *lib* y dentro de esta copiamos el contenido de la carpeta *target* donde están las librerías que se necesitan. Como se muestra en las siguientes figuras procedemos a vincular estas librerías a nuestro proyecto, tanto el SDK de Bluevia como sus dependencias:



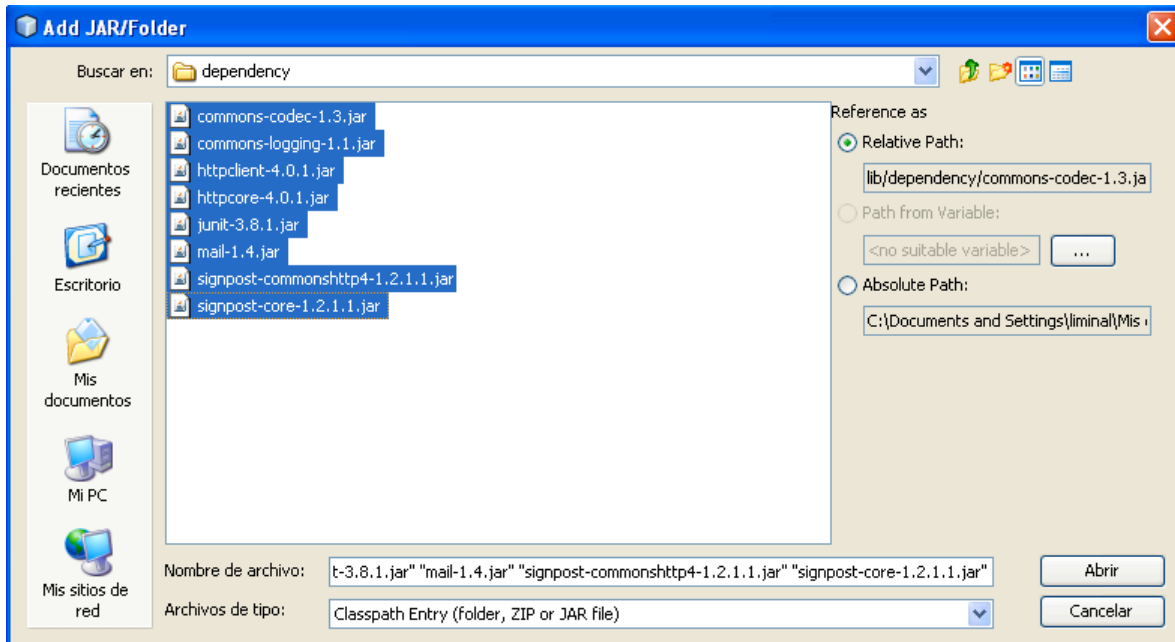


Figura 2 Agregar los JAR al proyecto

Con esto ya se puede comenzar con el desarrollo de aplicaciones en Bluevia. A continuación se presenta el proceso de registro en Bluevia.

5. REGISTRO EN BLUEVIA

Para utilizar el SDK, y por ende el API, de Bluevia es necesario registrarnos como desarrolladores dentro de la plataforma de Bluevia. Esto nos va a dar acceso a la creación de API Keys y a crédito en SMS y MMS para las pruebas que se necesiten. Para esto ingresamos a la página de [registro de Bluevia](#) e ingresamos la información que se solicita en el formulario, como se muestra en la Figura 3.

Register

Welcome!

To get up and running with BlueVia just complete this simple registration form. Once you have completed the form, click 'join' and we will send you an email with a confirmation link. Check your inbox and click this link to complete the registration process.

We hope you enjoy BlueVia, and be sure to let us know what you think.

Login Information

Email Address

Username

Choose a username of at least 5 and no more than 30 characters.


Password

Choose a password of at least 8 and no more than 30 characters.
At least one number is required.

Confirm Password

Figura 3 Formulario de Inscripción en Bluevia

Una vez completado el formulario el sistema nos solicitará que ingresemos nuestras credenciales para finalizar el proceso de inscripción:

 Please login to confirm your account

Log in

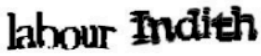
Username or Email




Password


[Did you forgot your password?](#) [Click to Register](#)

Keep me signed in for 2 weeks
Don't tick this box if you are using a public or shared account

Please resolve the captcha to proceed.

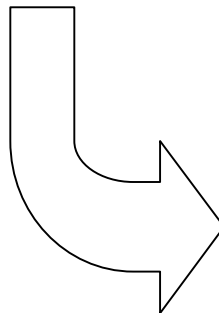
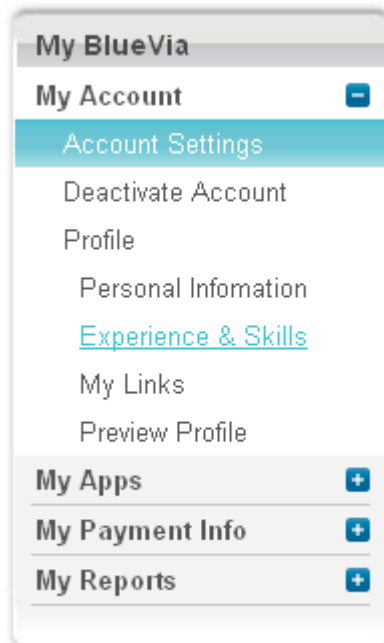
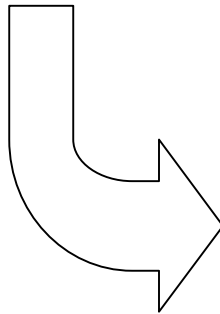
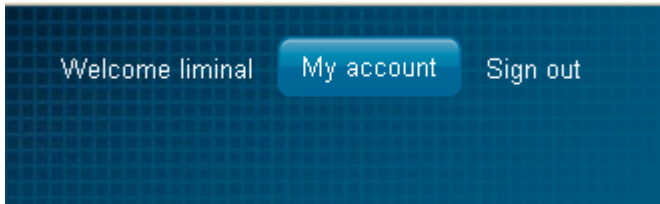





stop spam.
read books.

CancelLog In

Ya dentro de la plataforma es necesario activar nuestro número de celular para que quede vinculado a la cuenta y se pueda hacer uso de los créditos para probar. Para este registro seguimos el proceso descrito en la



Mobile Phone Number

Please include your country code e.g 447654320110

[Change password](#)

Es importante que al número de celular se le anteponga el código de país 57, un ejemplo de cómo debería ir el número es: **573181234567**.

Activación de cuenta de cliente

La creación de la cuenta de desarrollador automáticamente nos permite el acceso como clientes para poder probar nuestros desarrollos. Para activar esta cuenta debemos ir al sitio de [clientes de Bluevia](#). En esta página nos registramos con las MISMAS credenciales que creamos en el paso anterior y con este solo ingreso ya queda activada la cuenta. Además esta activación nos da 50 créditos para probar nuestras aplicaciones, como se presenta en la Figura 4

[My Account](#) ▶ My Service Credits

Service Bundles

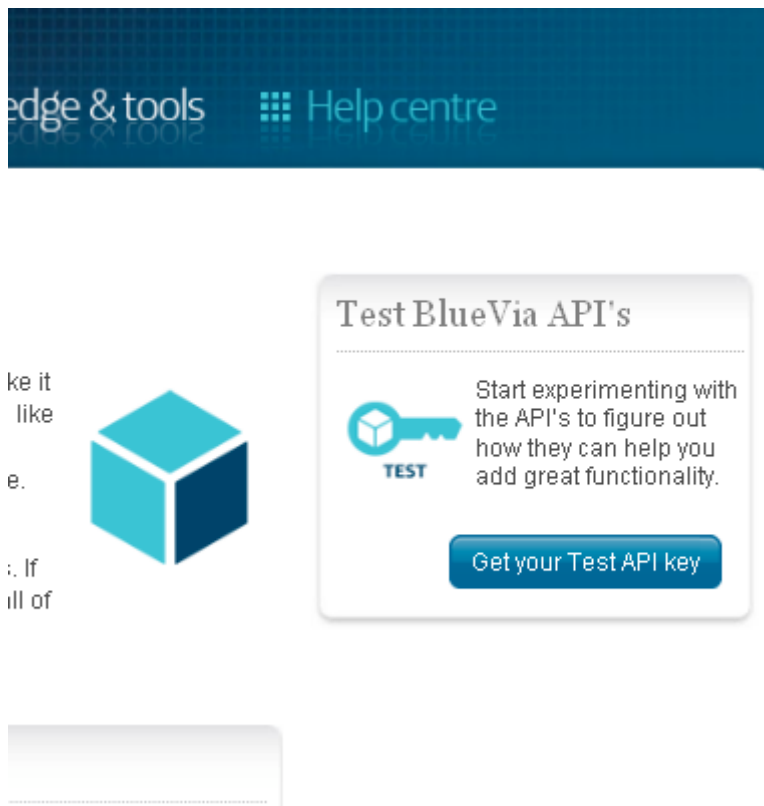
Manage your BlueVia Service Credits with BlueVia Connect. You can check your remaining units, upgrade to another BlueVia Service Credit, everything can be done easily with just a few clicks.

Coupon title	Type	Purchase Date	Expiry Date	Remaining
Free Test Credits. MMS Receive	MMS MO	13/06/2011	13/07/2011	50 units
Free Test Credits. MMS Send	MMS MT	13/06/2011	13/07/2011	50 units
Free Test Credits. SMS Receive	SMS MO	13/06/2011	13/07/2011	50 units
Free Test Credits. SMS Send	SMS MT	13/06/2011	13/07/2011	50 units

Figura 4 Créditos otorgados para pruebas

Obtención de API Keys

El último paso para comenzar a crear nuestras aplicaciones Java es la obtención de Bluevia API Keys. Para esto volvemos al sitio de Bluevia y en la [Sección de API](#) hacemos clic en *Get your Test API Key*.



Esto nos presentará un formulario, como el de la Figura 5, que debemos llenar con la información de nuestra aplicación.

Get your API key



Tell us about your application

Complete the description of your application below and we'll generate an API key to be used with your application. Learn more reading [BlueVia APIs guides](#).

Application name (must be unique)

Please tell us the name of your application

Description

Please describe your application

Language

Please choose your application language

APIs

- Advertising API
 - User Context API
 - Receive SMS
 - Send SMS API
 - Receive MMS
 - Send MMS API
 - Location API

Please choose the APIs your application is using

Figura 5 Formulario para obtener un API Key

Dependiendo de las API que se escojan se deberá ingresar un MO Keyword, este es un código alfanumérico permitirá reconocer que clientes nos han enviado mensajes a través de Bluevia. Este código deberá ser único y como sugerencia deberá ser informativo para mejorar el uso por parte de los clientes.

MO keyword

Enter your application keyword if you want to use the MO API. Test API keys will begin with 'TEST' as prefix but you don't have to write it.

Una vez ingresada la información el sistema genera un API key, como se presenta en la Figura 6.

Your API key

API key type	TESTING
Key	<input type="text" value="rJ11061379317903"/>
Secret	<input type="text" value="uQIX53447553"/>
Request Token URL	<input type="text" value="https://api.bluevia.com/services/REST/Oauth/getRequestToken/"/>
Access Token URL	<input type="text" value="https://api.bluevia.com/services/REST/Oauth/getAccessToken/"/>
Authorise URL	<input type="text" value="https://connect.bluevia.com/authorise/"/>
Application ID	956511d9fe0202bfe7aa52c7a2263865
Application name	liminal Test
Application description	Prueba para el taller de CP
Application language	SPA
API usage	Advertising API, User Context API, SMS, MMS, Location API
MO keyword	TESTLIM
Sandbox MO keyword	SANDLIM
adSpace	10977

Please do not forget to take a look at [BlueVia API guides](#) to get full information about how to make the requests and read the [BlueVia API reference](#) to get the endpoints.

[Go to My API keys](#)

Figura 6 Claves de acceso para una aplicación en Bluevia

Con el Key y Secret proporcionados es posible construir aplicaciones que utilicen la plataforma de Bluevia. Una vez completados todos estos pasos damos comienzo al desarrollo de dichas aplicaciones.

6. AUTENTICACIÓN Y PERMISOS

Bluevia es una plataforma que nos permite acceder a millones de clientes a través de aplicaciones. Para poder llegar a estos clientes es necesario que ellos nos “abran la puerta”, es decir, nos permitan utilizar su teléfono móvil y acceder a través de este a nuestras diversas aplicaciones. Para lograr este permiso Bluevia utiliza el protocolo OAuth, descrito ya en las charlas previas a este taller. Para autenticar a un usuario se deben seguir 5 pasos utilizando el SDK:

- 1. Definición del ambiente y de la información para obtener el Request Token:** El ambiente define si se va trabajar en modo Live o en modo Sandbox y la información necesaria es nuestro Consumer key y Consumer token (que lo obtenemos de la información provista por el API key).

Para manipularlos mejor dentro de la clase que utilizamos debemos definir dos variables que contengan esta información, como se muestra a continuación:

```
public static Mode mode = Mode.LIVE;
public static OAuthToken consumer = new OAuthToken("ConsumerKey",
"ConsumerToken");
```

Para nuestro ejemplo utilizaremos el modo LIVE, es decir, estaremos en capacidad de enviar mensajes a través de la red celular. Se deben reemplazar los términos ConsumerKey y ConsumerToken por los valores correspondientes de cada usuario.

- 2. Obtención del Request Token:** Para obtener este token se debe instanciar un objeto de tipo *RequestToken* e invocar el método *getRequestToken* para obtener nuestro request token. Este método, como se muestra en el ejemplo siguiente, recibe tres parámetros, el primero es nuestra información de autenticación de aplicación, el segundo es un Nick de usuario, que en nuestro caso no se usa, y el tercero una dirección web a donde redirigir la respuesta, en este ejemplo no estamos trabajando con dicha dirección por lo que se debe dejar nula. Este código ya debe estar incluido, en este ejemplo, dentro del método *main* del aplicativo.

```
RequestToken rt = new RequestToken();
OAuthToken requestToken = rt.getRequestToken(consumer, null, null);
System.out.println("Portal url:" + requestToken.getUrl());
```


En este ejemplo solo se utiliza la información de registro de la aplicación, por lo que es necesario que el usuario copie el vínculo en un browser. Para autorizar la aplicación el usuario debe ingresar sus credenciales, si no está registrado previamente, en el sitio (para nuestro caso las mismas credenciales de desarrollador) y le aparecerá una ventana como la de la Figura 7.

Confirmation required

This application is requesting to use BlueVia services on your behalf. To allow this you must provide permission, called "authorising"

Below you will see the services the application wishes to use.

Application Details



Application Name: liminal Test

Premium Services:

- SMS
- MMS API
- Directory API
- Advertising API
- LOCATION API

* This application is in test process and it will not generate charges. If you have spent all your free testing credits, you to wait until next month.

I have read and accept the [terms and conditions](#) and the [BlueVia privacy policy](#)

Cancel Authorise

Figura 7 Autorización de acceso

Paso seguido autorizamos a la aplicación y nos debe aparecer una respuesta como la de la Figura 8.

App Services Activated Successfully



App confirmation: Success activating application .

Please note you have to manually enter the verification code shown above on your BlueVía App before you can start use the application services.

You have successfully authorized the application to use the services.

Please remember that you can check and manage these permissions anytime at connect.bluevia.com.

Print Page

Application Details

Application Name: liminal Test

Premium Services: SMS
MMS API
Directory API
Advertising API
LOCATION API

Activation Details

PIN Code: 239778

Figura 8 Activación de servicios

En esta pantalla el sistema muestra un PIN, que es el código de autorización requerido en el siguiente paso.

- Solicitud de PIN:** El PIN *oauth verifier* que se obtuvo en el paso anterior es la entrada necesaria para obtener el Access Token. Para esto sencillamente le solicitamos el número al usuario:

```
Scanner in = new Scanner(System.in);
String oauth_verifier = in.nextLine();
```

- Obtención del Access Token:** Ya tenemos todos los elementos necesarios para solicitar el Access Token y finalizar con el proceso de autorización. Para esto creamos un objeto de tipo *AccessToken* e invocamos el método *get* para obtener nuestro Access Token. Este método recibe tres parámetros, el primero es nuestra información de autenticación de aplicación, el segundo el Request Token y el tercero el PIN que ingreso el usuario.

```
AccessToken at = new AccessToken();
OAuthToken accessToken = at.get(consumer, requestToken, oauth_verifier);
System.out.println("Access Token:" + accessToken.getToken());
System.out.println("Access Token Secret:" + accessToken.getSecret());
```

El resultado de este proceso se debería almacenar en una base de datos ya que este Access Token será utilizado todo el tiempo en los servicios de Bluevia. En nuestro caso debemos copiarlo en un

archivo aparte para tenerlo presente en los siguientes desarrollos de este taller. Una vez completado este paso podemos comenzar a recibir y enviar mensajes a nuestros clientes.

7. ENVIANDO Y RECIBIENDO MENSAJES

Con la autorización previa ya es posible enviar mensajes en nombre de nuestros clientes o recibir mensajes a nombre de ellos. Para facilitar las cosas es bueno crear un Access Token como elemento de la clase donde estamos trabajando, con la información obtenida:

```
public static OAuthToken accesstoken = new OAuthToken("AccessToken",
"AccessTokenSecret");
```

Se debe reemplazar AccessToken y AccessTokenSecret por la información obtenida en el numeral anterior. Con esta información procedemos a enviar nuestro primer SMS.

Envío de SMS

El envío de mensajes de texto utilizando la plataforma de Bluevia se realiza en tres sencillos pasos:

1. **Definición del mensaje:** Como primera medida se debe instanciar un objeto de tipo *com.bluevia.java.sms.MessageMT* y luego definir los números a donde se va a enviar el mensaje y el mensaje en sí, como se muestra a continuación:

```
MessageMT messageSender = new MessageMT(consumer, accesstoken, mode);
String addresses[] = { "573181234567" };
String message = "Hola Mundo Bluevia!";
```

El constructor del mensaje recibe tres parámetros, el primero es nuestra información de aplicación, el segundo es el Access Token de nuestro cliente y el tercero es el modo del ambiente. Los números de celular a donde se van a enviar el mensaje están contenidos en un arreglo, cada número debe iniciar con el código de país, en nuestro caso 57. El mensaje es una cadena de caracteres.

2. **Enviando el mensaje:** Con la información previamente definida se procede a enviar el mensaje. Para esto se invoca el método *send* con los números de celular y el mensaje como parámetros.

```
String smsId = messageSender.send(addresses, message);
```

Este método retorna una cadena con el número de identificación del mensaje enviado. Este número es importante para la posterior verificación del estado del mismo.

3. **Verificando el estado del mensaje:** Cuando el mensaje ha sido enviado debe ser verificado su estado para saber si llegó o no. Para esto utilizamos el identificador de mensaje y la función *getStatus*. Esta función nos permite verificar el último estado del mensaje, los diferentes mensajes que se pueden obtener se encuentran en la [página de Bluevia](#). El estado del mensaje es

generado en una lista por lo que en el ejemplo siguiente utilizamos un *foreach* para imprimir estos estados.

```
SMSDeliveryStatusType response = messageSender.getStatus(smsId);
List<DeliveryInformationType> list = response.getSmsDeliveryStatus();
for (DeliveryInformationType status : list) {
    System.out.println("Delivery status: " + status.getDeliveryStatus());
}
```

Este último paso nos permite verificar el mensaje y debería repetirse hasta que se reciba un estado de entrega satisfactoria del mensaje. Ahora procedemos a revisar la recepción de mensajes de texto.

Recepción de SMS

Si nuestro API key permite la recepción de SMS debimos haber establecido un MO Keyword, esta palabra nos permitirá recibir mensajes a través de la plataforma de Bluevia. Para recibir mensajes debemos efectuar tres pasos:

1. **Enviar un mensaje:** Un cliente puede enviar SMS a nuestra aplicación, para esto deberá enviar un mensaje al número 2505, que es el código de Bluevia especial para Colombia. La primera palabra del mensaje deberá ser el MO Keyword tal y como se escribió en el registro de la aplicación. Luego de esta palabra se debe escribir un espacio y ya el mensaje que se desee enviar como tal. Esto permitirá a Bluevia direccionar el mensaje a nuestra aplicación para que luego podamos recuperarlo.
2. **Definiendo elementos para la recuperación de mensajes:** Para que podamos recuperar nuestros mensajes necesitamos instanciar un objeto de tipo *com.bluevia.java.sms MessageMO* y definir el código de acceso a nuestros mensajes. La instancia de dicho objeto se hace invocando al constructor de la clase con tres parámetros ya previamente definidos como nuestra información de aplicación, el Access Token y el modo de trabajo. Además también debemos definir una cadena con el código de acceso a mensajes. Este código es el mismo al cual nuestros clientes envían los mensajes antecedido por el código del país, en nuestro caso será 572505.

```
MessageMO smsReceiver = new MessageMO(consumer, accesstoken, mode);
String registrationId = "572505";
```

3. **Recuperando los mensajes:** Para recuperar los mensajes se debe invocar el método *getMessages* y recorrer la lista que este retorna. Dicho método recibe como parámetro el código de acceso previamente definido. En el ejemplo siguiente se recorre la lista de mensajes utilizando un *foreach*. Es de tener en cuenta que una vez los mensajes han sido recuperados estos son borrados de la plataforma por lo que es necesario almacenarnos para su posterior utilización.

```

ReceivedSMSType response = smsReceiver.getMessages(registrationId);
List<SMSMessageType> list = response.getReceivedSMS();
for (SMSMessageType sms : list){
    System.out.println("Received SMS from '" +
        sms.getOriginAddress().getPhoneNumber() + "'>" + sms.getMessage() + "'");
}

```

A este método de recibir mensajes se le conoce como Polling y es uno de los dos métodos permitidos para este fin. El otro método es la notificación automática de mensajes. Este método permite que Bluevia nos notifique automáticamente cuando un mensaje es recibido. Se requiere una dirección web con soporte y certificado de HTTPS para recibir dichas notificaciones. Dado el alcance de este taller no se presenta este proceso pero los participantes pueden buscar la información pertinente en la página de Bluevia.

Envío de MMS

De igual manera que se envían SMS es el envío de MMS. La única diferencia relevante es el uso de adjuntos para estos mensajes. Se siguen los mismos tres pasos descritos en el envío de SMS. Es importante que el celular desde donde enviamos o a donde enviamos MMS debe tener configurado el acceso a redes de datos, sin esta configuración no se puede utilizar este servicio.

- 1. Definición del mensaje:** Como primera medida se debe instanciar un objeto de tipo *com.bluevia.java.mms.MessageMT* y luego definir los números a donde se va a enviar el mensaje y el mensaje en sí, como se muestra a continuación:

```

MessageMT messageSender = new MessageMT(consumer, accesstoken, mode);
String addresses[] = { "573181234567" };
Attachment[] files = {new Attachment("src/imagen.jpg", ContentType.IMAGE_JPEG),
    new Attachment("src/message.txt", ContentType.TEXT_PLAIN)};
String subject = "Enviando MMS!";

```

El constructor del mensaje recibe tres parámetros, el primero es nuestra información de aplicación, el segundo es el Access Token de nuestro cliente y el tercero es el modo del ambiente. Los números de celular a donde se van a enviar el mensaje están contenidos en un arreglo, cada número debe iniciar con el código de país, en nuestro caso 57. Luego definimos un arreglo de tipo *Attachment* con diferentes adjuntos que uno desee agregar. Cada uno de estos elementos debe incluir el path donde está el archivo, en nuestro caso debemos agregarlo en la carpeta src. Por último se define una cadena que contiene el asunto del mensaje a enviar

2. **Enviando el mensaje:** Con la información previamente definida se procede a enviar el mensaje. Para esto se invoca el método `send` con los números de celular, los adjuntos y el asunto del mensaje como parámetros.

```
String mmsId = messageSender.send(addresses, files, subject);
```

Este método retorna una cadena con el número de identificación del mensaje enviado. Este número es importante para la posterior verificación del estado del mismo.

3. **Verificando el estado del mensaje:** Cuando el mensaje ha sido enviado debe ser verificado su estado para saber si llegó o no. Para esto utilizamos el identificador de mensaje y la función `getStatus`. Esta función nos permite verificar el último estado del mensaje, los diferentes mensajes que se pueden obtener se encuentran en la [página de Bluevia](#). El estado del mensaje es generado en una lista por lo que en el ejemplo siguiente utilizamos un `foreach` para imprimir estos estados.

```
SMSDeliveryStatusType response = messageSender.getStatus(smsId);
List<DeliveryInformationType> list = response.getSmsDeliveryStatus();
for (DeliveryInformationType status : list) {
    System.out.println("Delivery status: " + status.getDeliveryStatus());
}
```

Este último paso nos permite verificar el mensaje y debería repetirse hasta que se reciba un estado de entrega satisfactoria del mensaje. Ahora procedemos a revisar la recepción de MMS

Recepción de MMS

Si nuestro API key permite la recepción de SMS debimos haber establecido un MO Keyword, esta palabra nos permitirá recibir mensajes a través de la plataforma de Bluevia. Para recibir mensajes debemos efectuar tres pasos:

1. **Enviar un mensaje:** Un cliente puede enviar MMS a nuestra aplicación, para esto deberá enviar un mensaje al número 2505, que es el código de Bluevia especial para Colombia. La primera palabra del mensaje deberá ser el MO Keyword tal y como se escribió en el registro de la aplicación. Luego de esta palabra se debe escribir un espacio y ya el mensaje que se desee enviar como tal. Esto permitirá a Bluevia direccionar el mensaje a nuestra aplicación para que luego podamos recuperarlo.
2. **Definiendo elementos para la recuperación de mensajes:** Para que podamos recuperar nuestros mensajes necesitamos instanciar un objeto de tipo `com.bluevia.java.mms.MessageMO` y definir el código de acceso a nuestros mensajes. La instancia de dicho objeto se hace invocando al constructor de la clase con tres parámetros ya previamente definidos como nuestra información de aplicación, el Access Token y el modo de trabajo. Además también debemos definir una

cadena con el código de acceso a mensajes. Este código es el mismo al cual nuestros clientes envían los mensajes antecedido por el código del país, en nuestro caso será 572505.

```
MessageMO smsReceiver = new MessageMO(consumer, accesstoken, mode);
String registrationId = "572505";
```

3. **Recuperando los mensajes:** Para recuperar los mensajes se debe invocar el método *getMessages* y recorrer la lista que este retorna. Dicho método recibe como parámetro el código de acceso previamente definido. En el ejemplo siguiente se recorre la lista de mensajes utilizando un *foreach*. Es de tener en cuenta que una vez los mensajes han sido recuperados estos son borrados de la plataforma por lo que es necesario almacenarlos para su posterior utilización.

```
ReceivedSMSType response = smsReceiver.getMessages(registrationId);
if (response == null || response.getReceivedMessages() == null
    || response.getReceivedMessages().isEmpty()){
    System.out.println("No messages");
} else {
    List<MessageReferenceType> list = response.getReceivedMessages();
    for (MessageReferenceType messageReference : list){
        ReceivedMMS mms = mmsReceiver.getMessage(registrationId,
            messageReference.getMessageIdentifier());
        System.out.println("Message from: " +
            mms.getMessage().getOriginAddress().getPhoneNumber());
        System.out.println(" > Subject: " +
            mms.getMessage().getSubject());
        ArrayList<MimeContent> contents = mms.getAttachments();
        for (MimeContent content : contents){
            System.out.println(content.getFileName());
        }
    }
}
```

A este método de recibir mensajes se le conoce como Polling y es uno de los dos métodos permitidos para este fin. El otro método es la notificación automática de mensajes. Este método permite que Bluevia nos notifique automáticamente cuando un mensaje es recibido. Se requiere una dirección web con soporte y certificado de HTTPS para recibir dichas notificaciones. Dado el alcance de este taller no se presenta este proceso pero los participantes pueden buscar la información pertinente en la página de Bluevia.

8. PUBLICIDAD

Por último vamos a evaluar la obtención de publicidad para nuestros clientes. Este servicio nos permite, de acuerdo al tipo de celular del cliente así como a sus preferencias, obtener publicidad que pueda llegar a su dispositivo. Para esto se debe utilizar la siguiente clase:

```
SimpleAd request = new SimpleAd();
request.setAdreqId("10654CC10-13-05T20:31:13c6c72731ad");
request.setUserAgent("Mozilla/5.0");
request.setAdSpace("10977");
request.setProtection_policy("1");
request.setCountry("CO");
Advertisement ad = new Advertisement(consumer, accesstoken, mode);
Creative_Elements response = ad.send(request);
```

La creación de la solicitud para un aviso publicitario necesita de varios elementos, en el ejemplo anterior se presentan estos elementos. El primero es un ID de aviso, que debe ser único y generado por nosotros, se recomienda utilizar una secuencia utilizando fechas, códigos de aplicación y algún tipo de encriptamiento. Luego se define el tipo de explorador en donde se presentará el aviso. A continuación se inicializa el AdSpace, este valor nos fue entregado con el registro del API key. Los últimos valores que se requieren son el nivel de protección (1 nivel alto baja contenido explícito – 3 bajo nivel de protección alto contenido explícito) y el país donde se desea mostrar el aviso. Con estos valores inicializados se solicita un nuevo aviso publicitario, el cual puede ser utilizado en nuestra aplicación o enviado al móvil de nuestro cliente. Para acceder a la información de este anuncio se puede utilizar el código presentado a continuación:

```
if (response != null) {
    for (Creative_Element e : response.getCreative_elements()) {
        System.out.println("creative_element_type_id: " + e.getType_id());
        System.out.println("creative_element_type_name: " + e.getType_name());
        System.out.println("creative_element_value: " + e.getValue());
        System.out.println("creative_element_interaction: " +
            e.getInteraction());
    }
}
```

9. EJERCICIO PRÁCTICO

El ejercicio práctico consiste en desarrollar un servlet-jsp sencillo que reciba mensajes SMS o MMS y los publique en una tabla. Una vez publicado deberá enviar un SMS de confirmación al número que envió el mensaje.