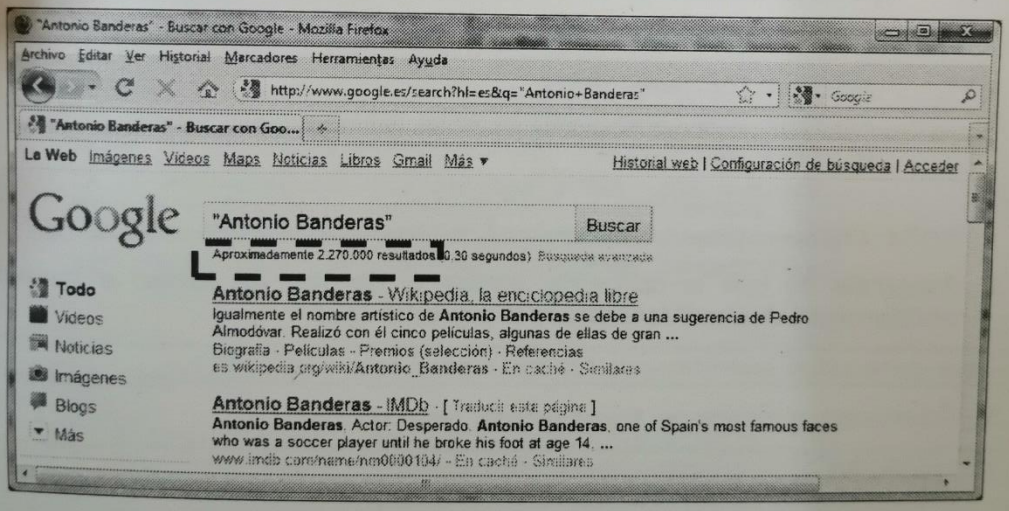
10.2.3. Utilizando HTTP desde Android

Tras el gran éxito de la web, el protocolo HTTP está siendo utilizado con finalidades diferentes de las que tenía en un principio: la descarga de páginas web. Por ejemplo, hoy en día es frecuente su uso para el intercambio de ficheros, la emisión de vídeo o la comunicación entre aplicaciones. A continuación describiremos las herramientas disponibles en Android para utilizar el protocolo HTTP. Para este propósito tenemos dos alternativas principales desde Android: el uso de las librerías `java.net.*` o `org.apache.commons.httpclient.*`. En el siguiente ejemplo utilizaremos las primeras.

En el ejemplo de este apartado vamos a extraer información de una de las páginas web más utilizadas en la actualidad: el servicio de búsqueda de Google. En concreto, nos interesa conocer el número de apariciones de una determinada secuencia de palabras en la web. En ciertas ocasiones esta información puede resultar muy interesante. Por ejemplo, tenemos dudas sobre el uso de una preposición en inglés: ¿se escribe *travel in bus* o *travel by bus*? Si buscamos ambas secuencias de palabras en Google, hay que ponerlas entre comillas para que busque la secuencia de forma literal. Obtenemos 240.000 apariciones para la primera y 1,1 millones para la segunda. Nuestra duda ha sido resuelta. Esta aplicación también puede utilizarse para averiguar quién es más popular en Internet, Antonio Banderas o Rafael Nadal.



Básicamente, la aplicación que mostramos a continuación funciona de la siguiente forma:

1. El usuario introduce una secuencia de palabras; por ejemplo, "Antonio Banderas".
2. Accedemos al servidor mediante la siguiente URL:

```
http://www.google.es/search?hl=es&q="Antonio+Banderas"
```

En este caso las peticiones se atienden mediante el método GET. En este método, si queremos enviar información al servidor hemos de incluirla tras un

carácter "?", seguido de un nombre de parámetro, seguido del carácter "=", seguido del valor. Los diferentes parámetros se separan mediante el carácter "&". Los espacios en blanco han de ser sustituidos por el carácter "+". En este ejemplo el parámetro *hl* corresponde al idioma de búsqueda y *q* a las palabras que hay que buscar.

3. Obtenemos la respuesta del servidor en una variable de tipo *string*.
4. Buscamos la primera aparición de "Aproximadamente" en la respuesta.
5. Tras esta palabra se encuentra la información que buscamos.

Obviamente, el correcto funcionamiento de esta aplicación está sujeto a que no se produzcan cambios en la forma en que Google visualiza los resultados. En caso de que se realice algún cambio en esta página, va a ser necesaria una adaptación de nuestra aplicación. La técnica de sacar información directamente de una página web se conoce como *web scraping*. Hoy en día existe otra alternativa más fiable para obtener información. En el siguiente apartado mostraremos cómo utilizar un servicio web con este propósito.

Ejercicio: *Búsquedas en Google con HTTP*

1. Crea una nueva aplicación con los siguientes datos:

```
Application name: HTTP
Package name: com.example.http
☑ Phone and Tablet
   Minimum SDK: API 9 Android 2.3 (Gingerbread)
```

***NOTA**: Utilizamos como versión mínima la 2.3 para poder configurar StrictMode.*

2. Asegúrate de que la aplicación solicita el permiso de acceso a Internet, añadiendo en *AndroidManifest.xml* la siguiente línea:

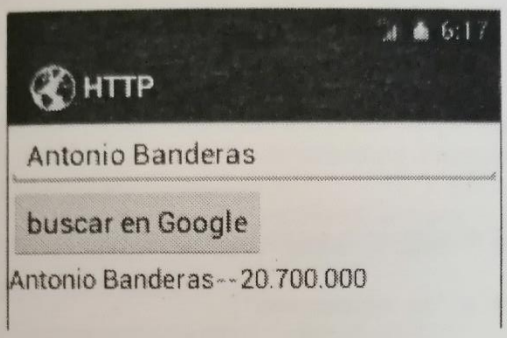
```
<uses-permission android:name="android.permission.INTERNET"/>
```

3. Reemplaza el código del *layout activity_main.xml* por:

```
<LinearLayout
   xmlns:android="http://schemas.android.com/apk/res/android"
   android:orientation="vertical"
   android:layout_width="match_parent"
   android:layout_height="match_parent">
   <EditText android:id="@+id/EditText01"
      android:layout_height="wrap_content"
      android:layout_width="match_parent"
      android:text="palabra a buscar"/>
   <Button android:id="@+id/Button01"
      android:layout_height="wrap_content"
      android:layout_width="wrap_content"
      android:onClick="buscar"
```

```
    android:text="buscar en Google"/>
  <TextView android:id="@+id/TextView01"
    android:layout_height="match_parent"
    android:layout_width="match_parent"
      android:textSize="8pt"/>
</LinearLayout>
```

La apariencia de este *layout* se muestra a continuación:



4. Reemplaza el código de *MainActivity.java* por:

```
public class MainActivity extends Activity {
   private EditText entrada;
   private TextView salida;

   @Override
   public void onCreate(Bundle savedInstanceState) {
      super.onCreate(savedInstanceState);
      setContentView(R.layout.activity_main);
      entrada = (EditText) findViewById(R.id.EditText01);
      salida = (TextView) findViewById(R.id.TextView01);
      StrictMode.setThreadPolicy(new StrictMode.ThreadPolicy.
            Builder().permitNetwork().build());
   }

   public void buscar(View view){
      try {
         String palabras = entrada.getText().toString();
         String resultado = resultadosGoogle(palabras);
         salida.append(palabras + "--" + resultado + "\n");
      } catch (Exception e) {
         salida.append("Error al conectar\n");
         Log.e("HTTP", e.getMessage(), e);
      }
   }

   String resultadosGoogle(String palabras) throws Exception {
      String pagina = "", devuelve = "";
      URL url = new URL("http://www.google.es/search?hl=es&q=\""
            + URLEncoder.encode(palabras, "UTF-8") + "\"");
```

```
        HttpURLConnection conexion = (HttpURLConnection)
              url.openConnection();
        conexion.setRequestProperty("User-Agent",
              "Mozilla/5.0 (Windows NT 6.1)");
        if (conexion.getResponseCode()==HttpURLConnection.HTTP_OK) {
           BufferedReader reader = new BufferedReader(new
              InputStreamReader(conexion.getInputStream()));
           String linea = reader.readLine();
           while (linea != null) {
              pagina += linea;
              linea = reader.readLine();
           }
           reader.close();
           int ini = pagina.indexOf("Aproximadamente");
           if (ini != -1) {
              int fin = pagina.indexOf(" ", ini + 16);
              devuelve = pagina.substring(ini + 16, fin);
           } else {
              devuelve = "no encontrado";
           }
        } else {
           salida.append("ERROR: "
                 + conexion.getResponseMessage() + "\n");
        }
        conexion.disconnect();
        return devuelve;
     }
}
```

El comienzo del código ha de resultarte familiar, posiblemente hasta la última línea del método `onCreate()`. En esta línea se configura StrictMode[38] para que permita accesos a la red desde el hilo principal.

Pasemos a describir el método `resultadosGoogle()`. Este método toma como entrada una secuencia de palabras y devuelve el número de veces que Google las ha encontrado en alguna página web. Lo primero que llama la atención es el modificador `throws Exception`. Estamos obligados a incluirlo si utilizamos la clase `HttpURLConnection`. Este modificador obliga a utilizar el método dentro de una sección `try … catch …`. La razón de esto es que toda conexión HTTP es susceptible de no poder realizarse, por lo que tenemos la obligación de tomar las acciones pertinentes en caso de problemas.

Tras la declaración de variables, creamos la URL que utilizaremos para la conexión. El método `URLEncoder.encode()` se encargará de codificar las palabras en el formato esperado por el servidor. Entre otras cosas reemplaza espacios en blanco, caracteres no ASCII, etc. A continuación, preparamos la conexión por medio de la clase `HttpURLConnection`. Mediante el método

`setRequestProperty()` podemos añadir cabeceras HTTP. En el punto 7 de este ejercicio se demuestra la necesidad de insertar la cabecera `User-Agent`.

En la siguiente línea se utiliza el método `getResponseCode()` para establecer la conexión. Si se establece sin problemas (`HTTP_OK`) leemos la respuesta línea a línea, concatenándola en el *string* `pagina`. De esta forma podemos buscar en `pagina` la primera aparición de la palabra "Aproximadamente". Si la encontramos, buscamos el primer espacio detrás del número que ha de aparecer tras "Aproximadamente" y devolvemos los caracteres entre ambas posiciones. En caso de no encontrar "Aproximadamente", puede que la secuencia buscada no se haya encontrado, aunque también es posible que Google haya cambiado la forma de devolver los resultados. En el caso de que la conexión no haya sido satisfactoria, devolvemos el mensaje de respuesta que nos dio el servidor `getResponseMessage()`.

5. Ejecuta la aplicación y verifica que funciona correctamente.
6. En el código anterior comenta la línea:

```
conexion.setRequestProperty("User-Agent", ...);
```

7. Ejecuta el programa. Ahora el resultado ha de ser:

```
ERROR: Forbidden
```

En este caso, al tratar de establecer la conexión el servidor, en lugar de devolvernos el código de respuesta `200: OK`, nos ha devuelto el código `403: Forbidden`. ¿Por qué? La cabecera `User-Agent` informa al servidor de qué tipo de cliente ha establecido la conexión. Según se demuestra en este ejercicio, el servicio de búsquedas de Google prohíbe la respuesta a aquellos clientes que no se identifican.

8. Analiza el valor asignado a la cabecera "Mozilla/5.0 ...". Puedes comprobar que la información que estamos dando al servidor es totalmente errónea.
9. Modifica este valor por "Ejemplo de El gran libro de Android" y comprueba que el resultado es `403: Forbidden` ¿Por qué no quiere responder? La respuesta es que, desde finales de 2011, el servidor de Google exige que el tipo de navegador que se conecte sea conocido.

10.2.4. Uso de HTTP con AsyncTask

En el ejercicio anterior hemos configurado el modo estricto para que nos permita realizar accesos a la red desde el hilo principal. Aunque en la mayoría de los casos la interacción con el servidor es inferior a 2 décimas de segundo, podrían darse algunos casos en que la interacción fuera superior a un segundo (servidores sobrecargados o redes muy lentas). En estos casos, la interfaz de usuario permanecería bloqueada un tiempo excesivo.

En el capítulo 5 hemos aprendido a utilizar la clase `AsyncTask` para resolver estos problemas. El siguiente ejercicio nos muestra cómo puede utilizarse en este caso: