Introducción a Android

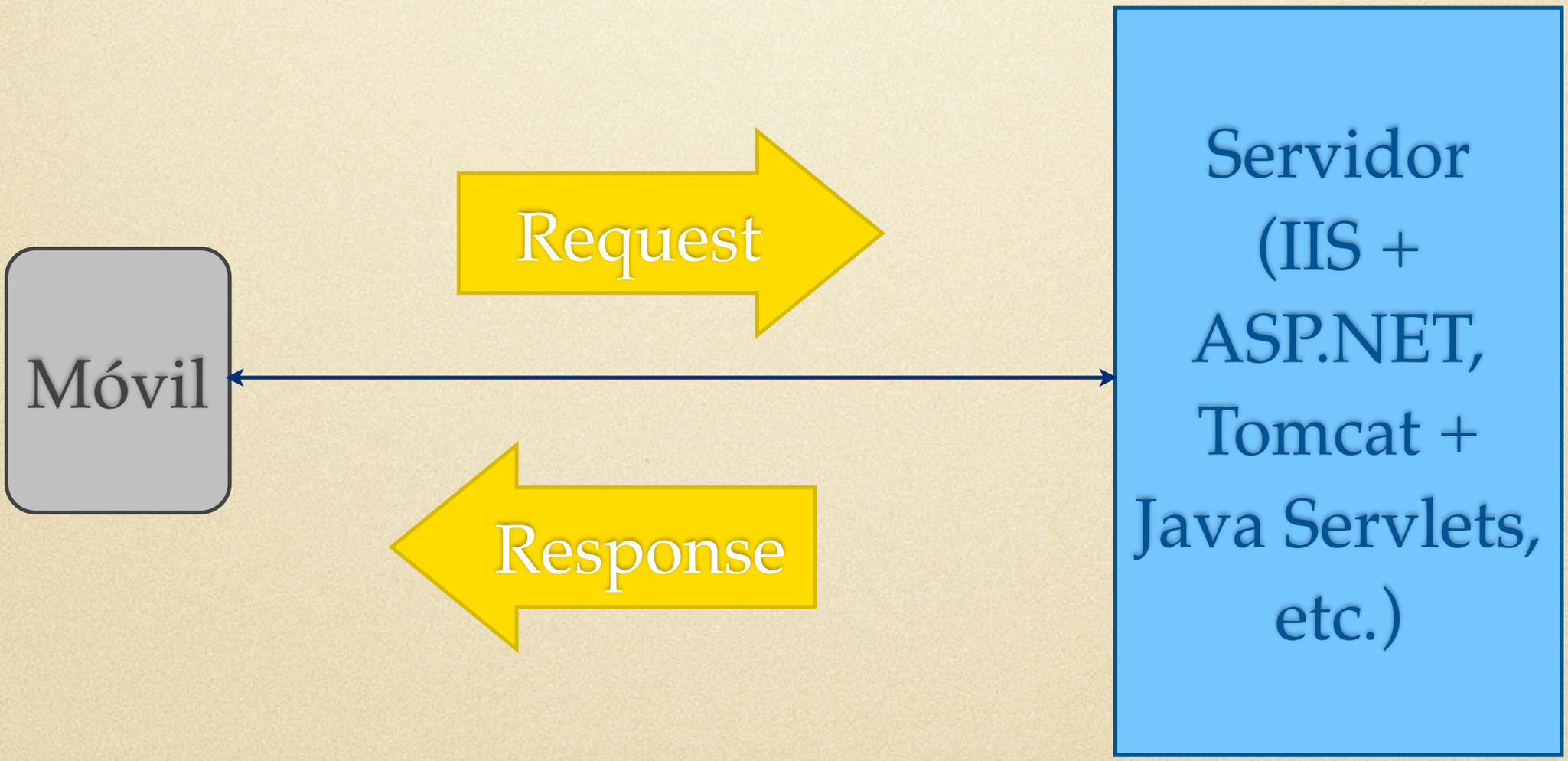
ANDROID

¿Qué vamos a ver?

- Conexión a un servicio HTTP

Conexión a la red / Internet

- Típicamente para obtener datos de una fuente externa o del servidor que almacena los datos:



Conexión a la red / Internet

- Siempre deben realizarse en un hilo separado, para evitar bloquear la UI
- HTTP es el protocolo más utilizado para esta comunicación. Android provee 2 clientes:
  - `HttpClient` de Apache: recomendado para versión 2.2 o anteriores
  - `HttpUrlConnection`: recomendado para versión 2.3 o posteriores

Implementación

- La aplicación debe tener los permisos adecuados en el archivo manifest:

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

- Opcionalmente, podemos verificar primero si hay conexión a la red:

```
ConnectivityManager connMgr = (ConnectivityManager)
    getSystemService(Context.CONNECTIVITY_SERVICE);
NetworkInfo networkInfo = connMgr.getActiveNetworkInfo();
if (networkInfo != null && networkInfo.isConnected()) {
    // obtener datos
} else {
    // mostrar error
}
```

Implementación

- Nuestro AsyncTask llama al método que hace la conexión en el `doInBackground`, y muestra los datos en el `onPostExecute` (que se ejecuta en el hilo de IU)

```
private class DownloadTask extends AsyncTask<String, Void, String> {

    @Override
    protected String doInBackground(String... arg0) {
        try {
            return descargarDesdeUrl(arg0[0]);
        } catch (IOException e) {
            return "No se pudo acceder a la URL " + arg0[0];
        }
    }

    @Override
    protected void onPostExecute(String result) {
        tviContenidoRed.setText(result);
    }
```

Implementación

- descargarDesdeUrl se encarga de crear la conexión HTTP:

```
private String descargarDesdeUrl(String strURL) throws IOException {
    InputStream is = null;
    try {
        URL url = new URL(strURL);
        HttpURLConnection conexion = (HttpURLConnection) url.openConnection();
        conexion.setReadTimeout(10000); // en milisegundos
        conexion.setConnectTimeout(15000); // en milisegundos
        conexion.setRequestMethod("GET");
        conexion.setDoInput(true);

        conexion.connect();
        int responseCode = conexion.getResponseCode();
        Log.d("EjemploConexion", "Codigo de respuesta HTTP: " + responseCode);
        is = conexion.getInputStream();

        String contenidos = leerDatos(is);
        return contenidos;
    }
    finally {
        if (is != null) {
            is.close();
        }
    }
}
```

Implementación

- Para leer los datos del `InputStream`:

```
private String leerDatos(InputStream stream) throws IOException {
    Reader reader = null;
    reader = new InputStreamReader(stream, "UTF-8");
    char[] buffer = new char[1024];
    StringBuffer bufferDatos = new StringBuffer();
    int cantCaracteresLeidos;

    while ((cantCaracteresLeidos = reader.read(buffer)) != -1) {
        bufferDatos.append(buffer, 0, cantCaracteresLeidos);
    }

    return bufferDatos.toString();
}
```

Librerías

- Volley
  - Mantenida por el Android Open Source Project
  - Múltiples características: programación automática de requests, conexiones de red concurrentes, caching, soporte para cancelación de requests

Librerías - Volley

```
final TextView mTextView = (TextView) findViewById(R.id.text);
...

// Crear el RequestQueue.
RequestQueue queue = Volley.newRequestQueue(this);
String url ="http://www.google.com";

// Solicitar una respuesta en texto de la URL proporcionada.
StringRequest stringRequest = new StringRequest(Request.Method.GET, url,
            new Response.Listener() {
    @Override
    public void onResponse(String response) {
        // Mostrar los primeros 500 caracteres de la respuesta.
        mTextView.setText("La respuesta es: "+ response.substring(0,500));
    }
}, new Response.ErrorListener() {
    @Override
    public void onErrorResponse(VolleyError error) {
        mTextView.setText("Error!");
    }
});
// Añadir el request al RequestQueue.
queue.add(stringRequest);
```

Librerías - Volley

- Cómo utilizarla:
  - Descargar / clonar el proyecto (https://android.googlesource.com/platform/frameworks/volley)
  - Añadir como proyecto a Eclipse y enlazar con nuestro proyecto de aplicación

Librerías

- AsyncHttp
  - Proyecto open-source independiente
  - Implementa las operaciones HTTP y los tipos de respuesta más comúnmente utilizados
  - Múltiples características adicionales: soporte para parámetros en el request, subida de archivos, soporte para cookies, etc.

Librerías - AsyncHttp

```
AsyncHttpClient client = new AsyncHttpClient();
client.get("http://www.google.com", new AsyncHttpResponseHandler() {

    @Override
    public void onStart() {
        // se ejecuta antes de iniciar el request
    }

    @Override
    public void onSuccess(int statusCode, Header[] headers, byte[] response) {
        // se ejecuta cuando la respuesta tiene status HTTP de "200 OK"
    }

    @Override
    public void onFailure(int statusCode, Header[] headers, byte[] errorResponse, Throwable e) {
        // se ejecuta cuando el status HTTP de la respuesta es "4XX" (ejm. 401, 403, 404)
    }

    @Override
    public void onRetry(int retryNo) {
        // se ejecuta cuando se reintenta el request
        }
});
```

Librerías - AsyncHttp

- Como utilizar:
  - Descargar desde http://loopj.com/android-async-http/
  - Incluir el archivo .jar dentro del proyecto