**Desarrollo de Aplicaciones para Dispositivos Móviles.**
**Tema**: Localización Geográfica.

# Parte I.
## Localización Geográfica.

En un móvil, la localización usualmente es con el GPS y con la red telefónica. Con el método `getAllProviders()` de la clase `LocationManager` se realiza la localización. Con una instancia del administrador de localización `getSystemService(LOCATION_SERVICE)` se obtiene la lista de proveedores del servicio:

```
LocationManager locManager       = (LocationManager)getSystemService(LOCATION_SERVICE);
List<String> listaProviders      = locManager.getAllProviders();
```

Con la generación de mensajes log se depuran las localizaciones de los proveedores de localización. Con una referencia al proveedor se invocan los métodos `getProvider(nombre)` y `getAccuracy()`, para la precisión `Criteria.ACCURACY_FINE` y `Criteria.ACCURACY_COARSE`, `supportsAltitude()` para la altitud, y `getPowerRequirement()` para el nivel de consumo de recursos del proveedor, es decir:

```
LocationManager locManager       = (LocationManager)getSystemService(LOCATION_SERVICE);
List<String> listaProviders      = locManager.getAllProviders();
LocationProvider provider        = locManager.getProvider(listaProviders.get(0));
int precision                    = provider.getAccuracy();
boolean obtieneAltitud           = provider.supportsAltitude();
int consumoRecursos              = provider.getPowerRequirement();
```

La clase `Criteria` indica la altitud y precisión:

```
Criteria req = new Criteria();
req.setAccuracy(Criteria.ACCURACY_FINE);
req.setAltitudeRequired(true);
```

Enseguida, se invocan los métodos `getProviders()` o `getBestProvider()` para la lista de proveedores que se ajusten al criterio definido. Por ejemplo:

```
String mejorProviderCrit = locManager.getBestProvider(req, false);//Mejor proveedor por criterio
List<String> listaProvidersCrit = locManager.getProviders(req, false); //Lista por criterio
```

La clase `LocationManager` posee al método llamado `isProviderEnabled()` al que se le pasa el nombre del proveedor. Para su localización:

```
LocationManager.NETWORK_PROVIDER        // para la red de telefónica.
LocationManager.GPS_PROVIDER            // para el GPS.
```

Para el GPS:

```
if (!locManager.isProviderEnabled(LocationManager.GPS_PROVIDER)){ //Si el GPS no está habilitado
      mostrarAvisoGpsDeshabilitado();
}
```

La clase `LocationManager` posee el método `getLastKnownLocation(String provider)`, que devuelve la última posición que se obtuvo a través del proveedor desde su uso. Se debe suscribir al evento que se lanza cuando un proveedor recibe nuevos datos sobre la localización actual.

Se le deben pasar cuatro parámetros al método `requestLocationUpdates()`:

- Nombre del proveedor de localización al que se desea suscribir.
- Tiempo mínimo, en milisegundos, entre actualizaciones.
- Distancia mínima, en metros, entre actualizaciones.
- Una instancia del objeto `LocationListener`, para definir las acciones de actualización de la posición.

La clase `LocationListener` posee métodos asociados a los eventos que se reciben del proveedor:

```
onLocationChanged(location)          para actualizar la posición.
```

`onProviderDisabled(provider)` cuando el proveedor se deshabilita.
`onProviderEnabled(provider)` cuando el proveedor se habilita.
`onStatusChanged(provider, status, extras)` cuando el proveedor cambia su estado. Sus valores pueden estar entre `OUT_OF_SERVICE`, `TEMPORARILY_UNAVAILABLE`, `AVAILABLE`.

Para implantar el método `onLocationChanged`, por ejemplo:

```
LocationListener locListener = new LocationListener() {
	public void onLocationChanged(Location location) {
		mostrarPosicion(location);
	}
	public void onProviderDisabled(String provider){
		lblEstado.setText("Proveedor en OFF");
	}
	public void onProviderEnabled(String provider){
		lblEstado.setText("Proveedor en ON");
	}
	public void onStatusChanged(String provider, int status, Bundle extras){
		lblEstado.setText("Status del Proveedor: " + status);
	}
};
```

Para visualizar la información de la posición se debe diseñar la interface XML como se indica en la figura siguiente.


Con el método `mostrarPosicion()` se muestran los datos de latitud, longitud y precisión:

```
private void mostrarPosicion(Location loc) {
	if(loc != null){
		lblLatitud.setText("Latitud: " + String.valueOf(loc.getLatitude()));
		lblLongitud.setText("Longitud: " +
		String.valueOf(loc.getLongitude()));
		lblPrecision.setText("Precision: " +
		String.valueOf(loc.getAccuracy()));
	}else{
		lblLatitud.setText("Latitud: (sin_datos)");
		lblLongitud.setText("Longitud: (sin_datos)");
		lblPrecision.setText("Precision: (sin_datos)");
	}
}
```

Para obtener la posición inicial y recibir la primera actualización con `LocationListener` al digitar el botón Activar:

```
private void comenzarLocalizacion(){
	locManager = (LocationManager)getSystemService(Context.LOCATION_SERVICE);
	Location loc = locManager.getLastKnownLocation(LocationManager.GPS_PROVIDER);
	mostrarPosicion(loc); //Obtenemos la última posición conocida y mostrarla
	locListener = new LocationListener() {
```

```
            public void onLocationChanged(Location location) {
                  mostrarPosicion(location);
            }
            :
      };
      locManager.requestLocationUpdates(LocationManager.GPS_PROVIDER, 30000, 0, locListener);
}
```

Las actualizaciones de la posición al GPS son cada 30 segundos. Para la desactivación:

```
btnDesactivar.setOnClickListener(new OnClickListener() {
      public void onClick(View v) {
            locManager.removeUpdates(locListener);
      }
});
```

Ahora, probar la aplicación en el emulador. Observar que no se posee acceso a un GPS.

# Parte II.

Revisar la forma de reducir el tiempo de espera y el comportamiento de la aplicación con mensajes log, los cuales se generan cuando:

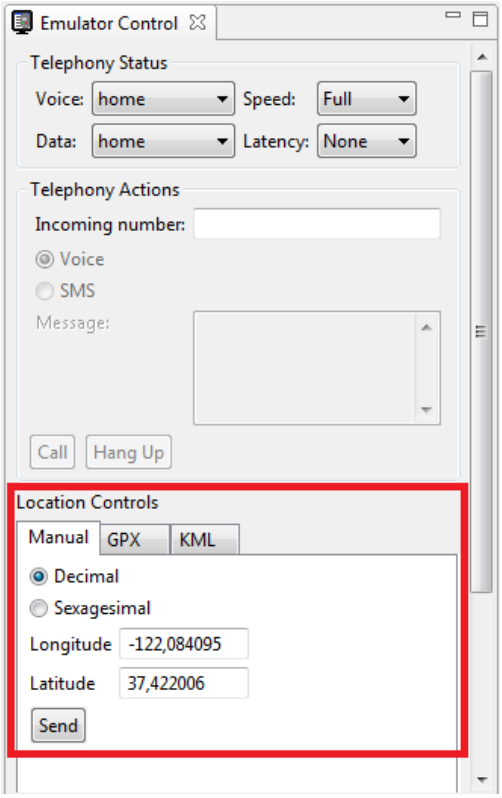
- El proveedor de localización cambia de estado y se muestra el nuevo estado.
- Se recibe una actualización de la posición y se muestran las nuevas coordenadas recibidas.

Es decir:

```
private void actualizarPosicion(){
      locationManager = (LocationManager)getSystemService(Context.LOCATION_SERVICE);
      Location location = locationManager.getLastKnownLocation(LocationManager.GPS_PROVIDER);
      muestraPosicion(location);
      locationListener = new LocationListener() {
            public void onLocationChanged(Location location) {
                  muestraPosicion(location);
            }
            public void onProviderDisabled(String provider){
                  lblEstado.setText("Provider OFF");
            }
            public void onProviderEnabled(String provider){
                  lblEstado.setText("Provider ON");
            }
            public void onStatusChanged(String provider, int status, Bundle extras){
                  Log.i("LocAndroid", "Provider Status: " + status);
                  lblEstado.setText("Provider Status: " + status);
            }
      };
      locationManager.requestLocationUpdates(
      LocationManager.GPS_PROVIDER, 15000, 0, locationListener);
}
private void muestraPosicion(Location loc) {
      if(loc != null) {
            lblLatitud.setText("Latitud: " + String.valueOf(loc.getLatitude()));
            lblLongitud.setText("Longitud: " + String.valueOf(loc.getLongitude()));
            lblPrecision.setText("Precision: " + String.valueOf(loc.getAccuracy()));
            Log.i("LocAndroid", String.valueOf(loc.getLatitude() + " - " +
                  String.valueOf(loc.getLongitude())));
      } else {
            lblLatitud.setText("Latitud: (sin_datos)");
            lblLongitud.setText("Longitud: (sin_datos)");
            lblPrecision.setText("Precision: (sin_datos)");
      }
}
```

Para inicializar una actualización de la posición se realiza un envío manual de una nueva posición al emulador de Android, para simular que se cambió la localización. En la pestaña Emulator Control del DDMS, en la sección Location

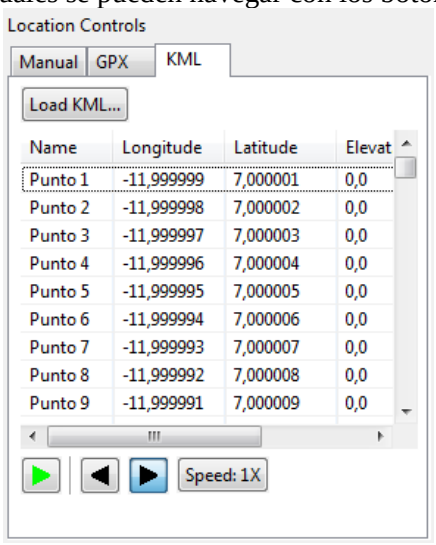
Controls. Introducir las coordenadas de longitud y latitud y digitar Send, ello ejecutará el evento `onLocationChanged()` y se mostrarán estos datos en la interfaz, como se indica enseguida:





También se puede proporcionar una lista de coordenadas con dos formas: en formato GPX o en archivo KML. En este caso se genera un archivo KML con 1000 posiciones geográficas aleatorias.

Acceder a Location Controls y digitar en la pestaña GPX o KML. Por ejemplo, para un archivo KML se digita en Load KML… y ver la lista de coordenadas, las cuales se pueden navegar con los botones incluidos:




# Ejecución de la aplicación.

En el emulador digitar Activar para detectar los cambios de posición y enseguida digitar en el botón verde. Observar que los valores de latitud y longitud se actualizan varias veces.

Permitir que durante un minuto se ejecute la aplicación y después detener la captura de posiciones digitando Desactivar.

Observar que en la ventana log del DDMS se muestran los mensajes de log. Se debe mostrar algo similar a lo siguiente:

```
05-08 10:50:37.921: INFO/LocAndroid(251): 7.0 - -11.999998333333334
05-08 10:50:38.041: INFO/LocAndroid(251): Provider Status: 2
05-08 10:50:38.901: INFO/LocAndroid(251): 7.000001666666666 - -11.999996666666668
05-08 10:50:39.941: INFO/LocAndroid(251): 7.000001666666666 - -11.999996666666668
05-08 10:50:41.011: INFO/LocAndroid(251): 7.000003333333333 - -11.999995000000002
05-08 10:50:43.011: INFO/LocAndroid(251): 7.000005000000001 - -11.999993333333334
05-08 10:50:45.001: INFO/LocAndroid(251): 7.000006666666667 - -11.999991666666665
05-08 10:50:46.061: INFO/LocAndroid(251): 7.000008333333333 - -11.999989999999999
05-08 10:50:47.131: INFO/LocAndroid(251): 7.000008333333333 - -11.999989999999999
05-08 10:50:47.182: INFO/LocAndroid(251): Provider Status: 1
05-08 10:51:02.232: INFO/LocAndroid(251): 7.000023333333333 - -11.999975
05-08 10:51:02.812: INFO/LocAndroid(251): 7.000023333333333 - -11.999973333333333
05-08 10:51:02.872: INFO/LocAndroid(251): Provider Status: 2
05-08 10:51:03.872: INFO/LocAndroid(251): 7.000024999999999 - -11.999973333333333
05-08 10:51:04.912: INFO/LocAndroid(251): 7.000026666666668 - -11.999971666666665
05-08 10:51:05.922: INFO/LocAndroid(251): 7.000026666666668 - -11.999971666666665
05-08 10:51:06.982: INFO/LocAndroid(251): 7.000028333333334 - -11.99997
05-08 10:51:08.032: INFO/LocAndroid(251): 7.000028333333334 - -11.999968333333333
05-08 10:51:09.062: INFO/LocAndroid(251): 7.00003 - -11.999968333333333
05-08 10:51:10.132: INFO/LocAndroid(251): 7.000031666666667 - -11.999966666666667
05-08 10:51:12.242: INFO/LocAndroid(251): 7.000033333333333 - -11.999965000000001
05-08 10:51:13.292: INFO/LocAndroid(251): 7.000033333333333 - -11.999963333333335
05-08 10:51:13.342: INFO/LocAndroid(251): Provider Status: 1
05-08 10:51:28.372: INFO/LocAndroid(251): 7.000048333333333 - -11.999950000000002
05-08 10:51:28.982: INFO/LocAndroid(251): 7.000048333333333 - -11.999950000000002
05-08 10:51:29.032: INFO/LocAndroid(251): Provider Status: 2
05-08 10:51:30.002: INFO/LocAndroid(251): 7.000050000000001 - -11.999948333333334
05-08 10:51:31.002: INFO/LocAndroid(251): 7.000051666666667 - -11.999946666666665
05-08 10:51:33.111: INFO/LocAndroid(251): 7.000053333333333 - -11.999944999999999
05-08 10:51:34.151: INFO/LocAndroid(251): 7.000053333333333 - -11.999944999999999
05-08 10:51:35.201: INFO/LocAndroid(251): 7.000055 - -11.999943333333333
05-08 10:51:36.251: INFO/LocAndroid(251): 7.0000566666666675 - -11.999941666666667
05-08 10:51:37.311: INFO/LocAndroid(251): 7.0000566666666675 - -11.999941666666667
05-08 10:51:38.361: INFO/LocAndroid(251): 7.0000583333333335 - -11.99994
05-08 10:51:38.431: INFO/LocAndroid(251): Provider Status: 1
```

Observar que se muestran los siguientes detalles:

- Un grupo inicial de actualizaciones entre las 10:50:37 y las 10:50:47, con 8 lecturas.
- Otro grupo de actualizaciones entre las 10:51:02 y las 10:51:13, con 11 lecturas.
- Un grupo más de actualizaciones entre las 10:51:28 y las 10:51:38, con 10 lecturas.
- Entre cada grupo de lecturas se transcurre un tiempo aproximado de 15 segundos.
- Los grupos se forman con un número variable de lecturas.

Buscar en el mensaje log que:

- Después de recibir cada grupo de lecturas, el proveedor pasa a estado 1 (`TEMPORARILY_UNAVAILABLE`).
- Después de iniciar la recepción de nuevas lecturas, el proveedor pasa a estado 2 (`AVAILABLE`).

**Nota: Generar un reporte completo con los archivos fuente y ejecutables obtenidos, incluyendo imágenes y conclusiones personales.**