Obtaining GPS fix using Java ME

There have been many posts on the Java Forum regarding the Location API (JSR-179) and the fact that sometimes no GPS fix can be obtained through those.

Many times the problem can be identified as a "time-out" exception, which means that the GPS chip couldn't return a GPS position within a determined time-window.

It has been apparent that some people actually try to obtain a GPS fix without knowing anything about GPS-technology, which makes it even harder to pinpoint where the problem might be...

So we are gonna make some assumptions and see where the developer himself could try to force a GPS fix out of this hard-headed hardware :D

The assumptions are :

- GPS-satellites are still happily circling the earth, and are all 100% functional, so no aliens actually shot those down
- the internal/external GPS-receiver is also a 100% functional, so the device didn't fall in water or something like that

So now you have written your code using the Location API's, and it runs pretty good in the emulator... But when installed on a device you simply can't seem to get a GPS fix... What can you do ?

Step 1

Most probably you have the Nokia Maps software installed on your device also. Fire that up and **WALK OUTSIDE** ! To get a GPS fix your device **must** be able to connect to at least 3 satellites, this **DOES NOT** happen if there is x inch/cm of concrete between the device and the outside air where the signals are actually traveling... So your device will need a good portion of clear sky to look for satellites.
If the Nokia Maps app will get a fix (note this may take even up to 5 minutes !!! This is called a cold-fix) you can close it down and fire up your own application. Hopefully this resolved your issue, otherwise let's try step 2.

Step 2

Well if step 1 didn't work and the aliens still haven't shot our GPS satellites down the problem might be in your code! At some point you will have set up a Criteria to be used to determine your LocationProvider. Try to "loosen" the criteria as much as possible, for example something like :

```
Criteria cr= new Criteria();
cr.setCostAllowed(true);
cr.setPreferredPowerConsumption(Criteria.NO_REQUIREMENT);
```

and maybe set the accuracy to very low, so setting a high value :

```
cr.setVerticalAccuracy(5000);
cr.setHorizontalAccuracy(5000);
```

Hopefully that helped ! Try this OUTSIDE once again :D

Step 3

Well, that didn't work ?. So last shot to get this problem out of the world! There are two methodologies to obtain a location with the Location API's :

Type 1 : Polling for a Location. You create a LocationProvider, feed it a Criteria and then you politely ask it to give you a Location :

```
lp= LocationProvider.getInstance(cr);
position = lp.getLocation(60);
```

Type 2 : Interrupt-driven. You still create a LocationProvider and feed it a Criteria. But then you make use of the LocationListener() Interface so that when a Location is available (or when you need a Location every x seconds) it will simply be passed to the LocationUpdated() method within your MIDlet!

First extend the class with the interface:

```
public class myClass implements LocationListener
```

Then attach the `LocationListener` to the `LocationProvider` :

```
lp.setLocationListener(this, interval, -1, -1);
```

And then obviously this method will be needed as part of the `LocationListener` interface:

```
public void locationUpdated(final LocationProvider locationProvider, final Location
location)
```

That last one will be called whenever a Location is available...
Also a complete code example can be find within :
Java Developers Library 3.3 -> JavaDocs -> (JSR-179)Location API's

There have been reports of problems whilst using the "polling" mechanism, so ALWAYS try the LocationListener mechanism as well !!!

Again *GO OUTSIDE* and test the app!

Hopefully now you are able to get a GPS fix, if not most probably the aliens actually did shoot down the satellites, and the last thing you should be worying about is getting a GPS fix!