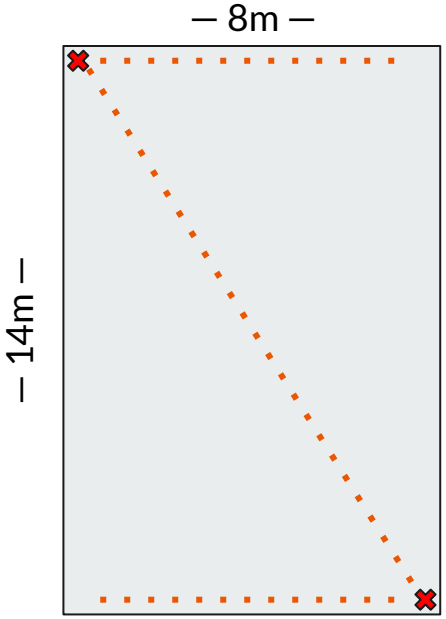
Présentation projet MI12

Recherche d’une balise Bluetooth cachée

Plan

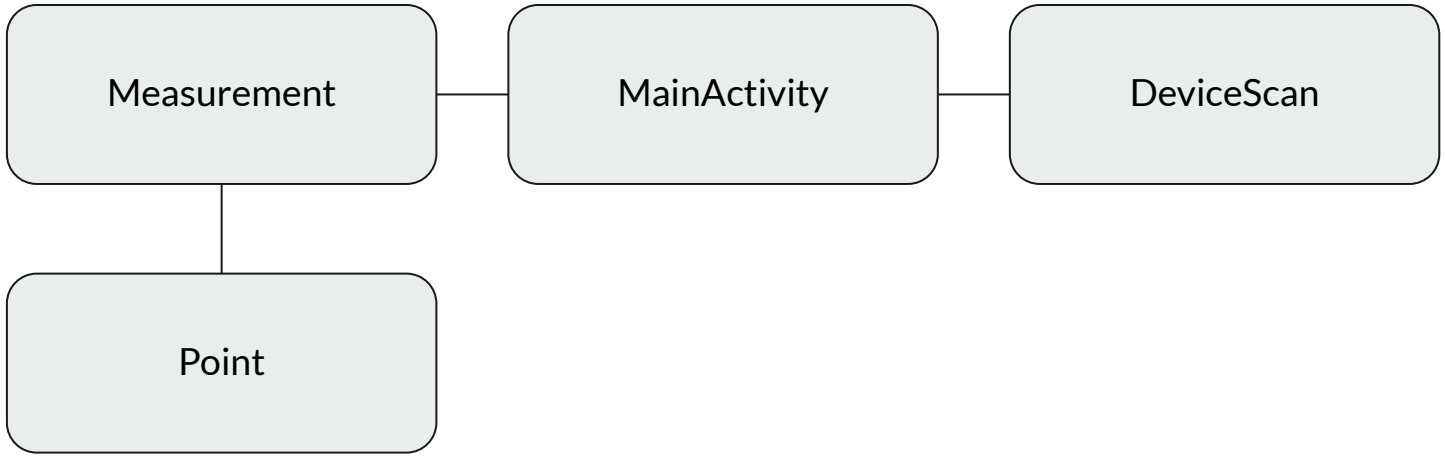
1. 1ère méthode (avec cartographie)
    a. Méthode cartographie
2. 2ème méthode (sans cartographie)
    a. Structure du programme
    b. Interface
    c. Interpolation linéaire
3. Conclusion et prise de recul

1ère méthode - cartographie



- 1 point par mètre
- 112 points
- Pour chaque point : RSSI moyen des balises connues
- tab = [
  [
  [Point 1, RSSI Balise1],
  [Point 1, RSSI Balise2],
  ……..
  ]
  ……
  ]

2ème méthode - sans cartographie



DeviceScanActivity

protected int rsiAverage

private List<Integer> rsiList1 = new ArrayList<>()

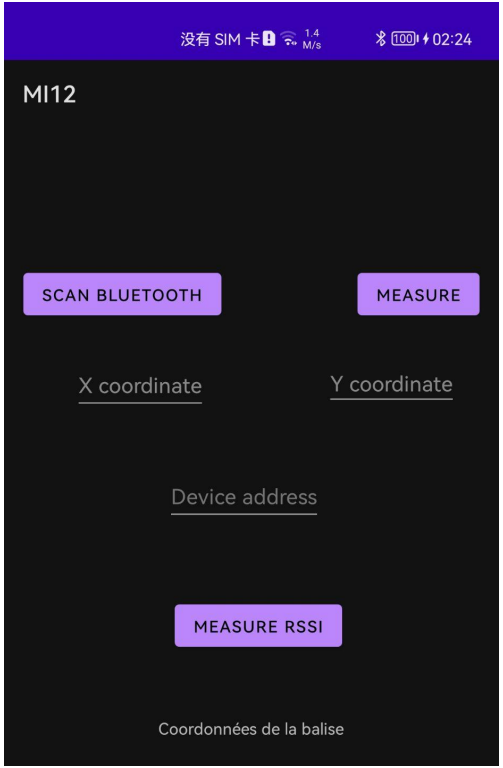
public double scanLeDevice(final boolean enable, String address, Point point)

public int getRsiAverage(String address)

Measurement

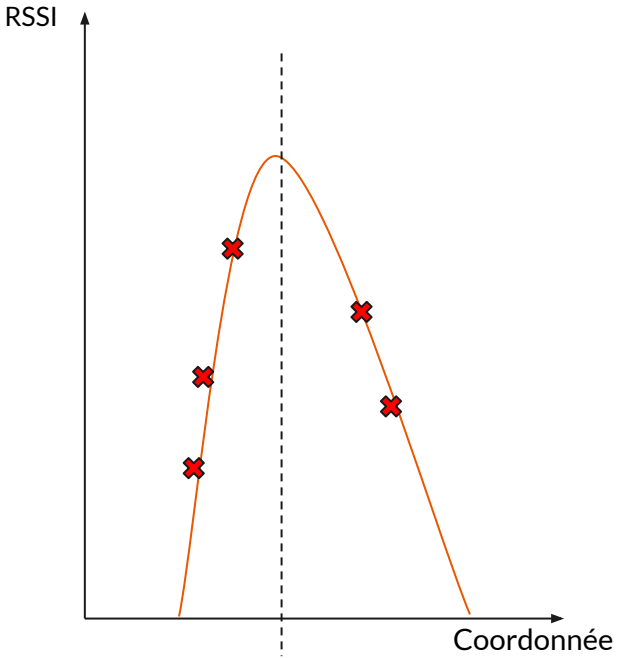
private Point point private double rssi
public static Point getPosition(List<Measurement> measurements) private static Point interpolate(Map<Point, Double> map, Point bestPoint)

Interface


没有 SIM 卡
1.4 M/s
02:24
MI12
SCAN BLUETOOTH
MEASURE
X coordinate
Y coordinate
Device address
MEASURE RSSI
Coordonnées de la balise

Interpolation linéaire

```
private static Point interpolate(Map<Point, Double> map, Point bestPoint) {
    Point p1 = new Point( x: 0, y: 0), p2 = new Point( x: 0, y: 0);
    double r1 = -100, r2 = -100;
    for (Map.Entry<Point, Double> entry : map.entrySet()) {
        Point point = entry.getKey();
        double rssi = entry.getValue();
        if (point.getX() < bestPoint.getX() || (point.getX() == bestPoint.getX() && point.getY() < bestPoint.getY())) {
            if (rssi > r1) {
                p1 = point;
                r1 = rssi;
            }
        } else if (point.getX() > bestPoint.getX() || (point.getX() == bestPoint.getX() && point.getY() > bestPoint.getY())
            if (rssi > r2) {
                p2 = point;
                r2 = rssi;
            }
        }
    }
    if (p1 == null || p2 == null || r1 == -100 || r2 == -100) {
        return bestPoint;
    } else {
        double x = (bestPoint.getX() * r2 - p2.getX() * r1) / (r2 - r1);
        double y = (bestPoint.getY() * r2 - p2.getY() * r1) / (r2 - r1);
        System.out.println("Interpolate X : " + x);
        return new Point(x, y);
    }
}
```



Conclusion

Xinghua Shao - Damien Vaurs