

第 16 章 定位

与云计算、大数据和物联网一样，LBS 已经从飘在空中的概念成功落地，渗透到人类生活的方方面面，一切服务都在基于位置。人们逛街购物、餐饮、娱乐游戏、工作学习、旅游出行、健康医疗、教育学习，均与地理位置紧密结合起来。毫不夸张地说，LBS 现在已经影响到每一个人，它就像空气和水一样成为必需品，无处不在。话说我前几天去太原和平遥，步行还用手机的地图通过 GPS 来定位酒店和景点……难道仅仅因为我是路痴吗……

移动端的所有应用几乎都需要读取用户位置信息：新闻客户端需要根据用户位置推送本地新闻；酒店应用需要根据用户位置搜索附近酒店；团购应用需要告诉用户附近优惠；打车应用需要知道所在的位置；手机游戏需要结合定位做一些线下交互……LBS 早已不是地图和导航的专利，而是成为移动互联网的一项基本能力，现在打开任何一个用户的手机，不读取位置权限的应用已属凤毛麟角很难找到了。

位置信息可能来自卫星（GPS）、WiFi 或者其他途径（比如移动基站）。

Qt Quick 提供了一组类库，让开发者可以获取位置信息，要使用它们需要引入 QtPositioning 模块，像下面这样：

```
import QtPositioning 5.2
```

16.1 类库介绍

QtPositioning 模块包含了很多类，比较重要的有全局的 QtPositioning 对象、PositionSource 类、Position 类以及 coordinate 类，和这 4 个小伙伴搞好关系，是开发基于位置的应用的前提。

想了解 QtPositioning 模块的所有类型，请以“Qt Positioning QML types”为关键字在帮助中检索。

16.1.1 coordinate

coordinate 是 QtPositioning 模块提供的基本类型，用来表示一个地理位置。

longitude、latitude、altitude 三个 real 类型的属性分别代表经度、纬度、高度。

布尔型的属性 `isValid`，提示位置信息是否有效。

`distanceTo()`方法可以计算当前位置与其他位置之间的距离，返回值为 `real` 类型，单位是米。函数原型：`real distanceTo(coordinate other)`。

`real azimuth(coordinate other)`方法返回与其他位置之间的方位角。从某点的指北方向线起，依顺时针方向到目标方向线之间的水平夹角，叫方位角。

`coordinate atDistanceAndAzimuth(real distance, real azimuth)`方法根据给定的距离和方位角，计算出对应的地理位置信息。

16.1.2 QtPositioning

`QtPositioning` 是个单例对象，它提供了一些方法，用于构造其他对象。

`coordinate coordinate(real latitude, real longitude, real altitude)`方法用于构建一个 `coordinate` 对象。

还有 `rectangle()`、`circle()`、`shape()`等方法，请查阅 Qt 帮助了解详情。

16.1.3 Position

`Position` 保存特定时间点的位置信息，比如坐标、速度等。

只读属性 `coordinate` 保存地理坐标。与其相关的有 `longitudeValid`、`latitudeValid`、`altitudeValid` 三个 `bool` 型属性，指示对应的值是否有效。

只读属性 `timestamp` 保存获取到位置信息的时间，类型是 `date`。

`direction` 属性保存与真北方向（某点指向北极的方向线叫真北方向线）之间的夹角。
`directionValid` 指示 `direction` 是否有效。

`real` 类型的 `horizontalAccuracy` 代表水平精度，单位是米。`horizontalAccuracyValid` 指示 `horizontalAccuracy` 是否有效。类似的还有 `verticalAccuracy`、`verticalAccuracyValid` 两个属性。

`speed` 属性表示对地速度，单位是米/秒，类型是 `double`，对应的有一个 `speedValid` 指示其有效性。`verticalSpeed` 表示垂直速度，对应的有 `verticalSpeedValid` 指示其有效性。

16.1.4 PositionSource

`PositionSource` 类是定位功能的核心哦，用于获取用户的位置信息，用起来很简单，只要设置其 `active` 属性为 `true`，实现 `onPositionChanged` 信号处理器即可。

示例代码片段：

```
PositionSource {
    updateInterval: 1000;
    active: true;

    onPositionChanged: {
        var coord = src.position.coordinate;
        console.log("Coordinate:", coord.longitude, coord.latitude);
    }
}
```

`active` 属性设置为 `true`，相当于调用 `start()` 方法，告诉 `PositionSource` 开始获取位置信息；设置为 `false`，相当于调用 `stop()` 方法，终止获取位置信息。与此相关的还有一个 `update()` 方法，用于获取一次位置信息。而 `start()` 方法调用后，`PositionSource` 会根据设置的 `updateInterval` 周期性地获取位置信息。

`position` 属性类型为 `Position`，保存了从 GPS 等来源获取的详细位置信息。

`name` 属性保存了正在提供位置服务的模块的名字。你也可以设置它来指定使用某个位置服务。

`valid` 属性指示 `PositionSource` 是否找到了可用的位置服务插件，如果没有找到，它的值为 `false`。

`supportedPositioningMethods` 属性保存当前位置信息源支持的定位方式，是个枚举值，可能的值有 `PositionSource.NoPositioningMethods`（不可用）、`PositionSource.SatellitePositioningMethods`（卫星定位）、`PositionSource.NonSatellitePositioningMethods`（支持非卫星定位）、`PositionSource.AllPositioningMethods`（卫星定位和非卫星定位都支持）。

`preferredPositioningMethods` 属性保存当前位置信息源的推荐定位方式，取值范围与 `supportedPositioningMethods` 一样。

16.2 团购查询实例

设计了一个简单的实例，通过 `QtPositioning` 获取到用户的地理位置，再调用百度 Geocoding API 查询经纬度对应的城市，最后调用百度 Place API 查询团购信息。

16.2.1 百度 API 说明

(1) 查询城市

根据经纬度查询城市，我使用 Geocoding API 中的“逆地理编码服务”，要求 API 返回 json 格式的数据。例如：

```
http://api.map.baidu.com/geocoder/v2/?ak=E4805d16520de693a3fe707cdc962045&location=39.983424,116.322987&output=json
```

我在程序中指定了 `location`（先纬度后经度，中间用逗号分隔）、`ak`（应用 key）、`output`（返回的数据格式，支持 xml、json 两种）三个参数，其实逆地理编码 API 支持更多的参数，详情请参考 <http://developer.baidu.com/map/webservice-geocoding.htm>。

(2) 查询团购

我使用 Place API 中的团购检索服务查询团购信息，目前支持餐饮、娱乐、旅游住宿、生活 4 个分类。试验后发现生活这个分类总查不到信息，实例中去掉了。

示例中是这么用的 API：

```
http://api.map.baidu.com/place/v2/eventsearch?query=%1&region=%2&event=groupon&location=%3,%4&output=json&page_size=6&ak=7fa612aa9f51f2dab4e685404afdcf25
```

其中 %1 为使用 `encodeURIComponent()` 方法编码过的城市名字；%2 为城市代码，如西安为 233；`location` 参数中的 %3 最终被扩展为纬度，%4 是经度；我指定返回数据格式为 json，

一页返回 6 个商家的数据。

到 <http://developer.baidu.com/map/webservice-placeapi.htm> 页面可以了解 Place API 的详情。

16.2.2 手机运行效果

因为手机的 GPS 功能在室内用不了，我就选择了一个大雨初停的下午，抱着笔记本电脑来到了小区内的一个凉亭内，喂了几十分钟的蚊子，带着满身的伤痕和难熬的痒与疼，终于截取了几张效果图，真不容易啊，入秋的蚊子可是真毒真疯啊。

图 16-1 是初始运行效果图（我裁切了图片）。



图 16-1 团购查询之获取位置

图 16-2 是查询到城市信息后的效果图。



图 16-2 团购查询之获取到城市信息

图 16-3 是查询餐饮和娱乐相关的团购后的效果图。



图 16-3 团购查询之餐饮和娱乐查询结果

本实例仅仅是为演示 QtPositioning API 而设计的，没有展示团购详情。你如果有兴趣，可以根据百度的 Place API 来扩展本实例。

16.2.3 源码分析

项目创建过程请参考 2.3 节，这里不再赘述。

代码的逻辑很简单：启动应用后读取位置信息，获取经纬度后调用百度的接口查询城市信息，拿到城市后保存下来，当用户点击某一类别（如餐饮）时调用团购 API 查询。

所有代码都在 main.qml 文件中，我们拆开来看看吧。

(1) 界面

图 16-1 和 16-2 显示的加载效果和提示，使用的是 BusyIndicator 和 Text 对象，Z 序设置为 2，比用于显示团购结果的 ListView 对象的 Z 序大。

如图 16-3 所示，顶部的分类、位置信息是通过 ListView 的 header 实现的。代码如下：

```
Component {
    id: actionView;
    Item {
        width: parent.width;
        height: 46;
        property alias area: location;

        Rectangle {
            id: splitter;
            color: "transparent";
            border.width: 1;
            height: 2;
            border.color: "#666666";
            width: parent.width;
            anchors.bottom: parent.bottom;
        }
    }
}
```

```
Button{
    id: eat;
    anchors.top: parent.top;
    anchors.topMargin: 2;
    anchors.left: parent.left;
    anchors.leftMargin: 4;
    anchors.bottom: splitter.top;
    style: flat;
    text: "餐饮";
    onClicked: searchGroupon(text);
}
Button{
    id: enjoy;
    anchors.top: eat.top;
    anchors.left: eat.right;
    anchors.bottom: eat.bottom;
    anchors.leftMargin: 4;
    style: flat;
    text: "娱乐";
    onClicked: searchGroupon(text);
}
Button{
    id: travel;
    anchors.top: eat.top;
    anchors.left: enjoy.right;
    anchors.bottom: eat.bottom;
    anchors.leftMargin: 4;
    width: 140;
    style: flat;
    text: "旅游住宿";
    onClicked: searchGroupon(text);
}

Text {
    id: location;
    anchors.leftMargin: 4;
    anchors.left: travel.right;
    anchors.top: travel.top;
    anchors.bottom: travel.bottom;
    font.pointSize: 15;
    color: "steelblue";
    verticalAlignment: Text.AlignVCenter;
}
}
```

actionView 中放了三个 Button、一个 Text，Button 通过 ButtonStyle 做了简单的定制。为了分隔分类和具体的 Item，我还在 actionView 底部放了一个只有边框的 Rectangle 对象。

团购查询出来的结果，是通过 ListView 的 Item 显示的，Item 的绘制由 ListView 的 itemDelegate 属性指向的组件完成，该组件的定义如下：

```
Component {
    id: businessDelegate;
    Item {
        id: wrapper;
        width: parent.width;
        height: 150;
        MouseArea {
            anchors.fill: parent;

            onClicked: wrapper.ListView.view.currentIndex = index;
        }
    }
}
```

```

Image {
    id: pic;
    x: 2;
    y: 2;
    width: 146;
    height: 146;
    source: picture;
}

ColumnLayout {
    anchors.left: pic.right;
    anchors.top: pic.top;
    anchors.right: parent.right;
    anchors.bottom: parent.bottom;
    anchors.margins: 2;
    spacing: 2;
    Text {
        Layout.fillWidth: true;
        font.pointSize: 15;
        font.bold: true;
        color: "white";
        text: "<b><font color='#0000FF'>%1</font></b>,<b>%2 个结果".arg(name).arg(count);
    }
    Text {
        Layout.fillWidth: true;
        font.pointSize: 12;
        color: "white";
        elide: Text.ElideRight;
        text: "地址:%1".arg(address);
    }
    Text {
        Layout.fillWidth: true;
        font.pointSize: 12;
        color: "white";
        text: "电话:%1".arg(phone);
    }
    Text {
        Layout.fillWidth: true;
        font.pointSize: 12;
        color: "white";
        text: "距离:%1 米".arg(distance);
    }
}
}

```

businessDelegate 把 Item 的界面分为两部分，一部分是左侧的 Image 对象，一部分是右侧的 4 行商家详情，我使用锚布局管理它们。而右侧的商家详情，又使用 ColumnLayout 来管理。

使用 MouseArea 来响应用户点击，在 onClicked 信号处理器内设置 ListView 的 currentIndex 属性为接收到点击的 Item 的 index，这样手指点了界面后，高亮背景就会移动了。

ListView 使用 ListModel，当查询到团购时，通过 append 来添加条目。具体代码如下：

```

function parseGroupon(jsonText){
    groupon = JSON.parse(jsonText);
    if(groupon == null || groupon.status != 0 || groupon.results.length == 0){
        clue.text = "抱歉哦";
    }else{

```

```
listview.model.clear();
clue.visible = false;

var it;
for(var i = 0; i < groupon.results.length; i++){
    it = groupon.results[i];
    listview.model.append(
        {
            "name": it.name,
            "address": it.address,
            "phone": it.telephone,
            "distance": it.distance,
            "count": it.events.length,
            "picture": it.events[0].groupon_image
        }
    );
}
}
busy.running = false;
}
```

给 model 添加数据时, 我使用对象的字面量表示法创建 ListElement 对象传递给 append() 方法。

(2) PositionSource 的使用

团购查询实例只需要用户的经纬度信息即可, 因而对 PositionSource 的使用也非常简单, 仅仅是实现了 onPositionChanged 信号处理器来响应位置变化。代码如下:

```
function gotPosition(){
    geopos = positionSource.position.coordinate;
    if(geopos.isValid && city == null){
        clue.text = "正在查询城市信息...";
        getCity();
    }else{
        console.log("update failed, wait...");
    }
}

PositionSource {
    id: positionSource;
    updateInterval: 1000;
    active: true;
    onPositionChanged: {
        root.gotPosition();
    }
}
```

(3) HTTP 下载与 JSON 解析

访问百度 API 时, 使用 XmlHttpRequest 类, 在 15.2 节介绍过。而 JSON 数据的解析, 则使用 ECMAScript 的内置对象 JSON, 一旦解析完成, 就可以把结果当作普通对象来访问, 只要对照着 API 定义的格式使用即可。

好啦, 代码就介绍到这里了, 完整的代码和项目在这里: <https://github.com/foruok/qtquick/NearService>。