

Application: Ad2Location

Company: UBIWIRELESS, LLC

Topic: Location-Based Services, Mobile Web

Author: Edwin Hernandez

Site: <http://android.ubiwireless.com/>

CONTENT

INTRODUCTION	1
APPLICATION INSTRUCTIONS	1
INSTRUCTIONS	1
AD2LOCATION CONCEPT	2
IMPLEMENTATION ASSUMPTION	3
XML DEFINITION LANGUAGE	3
TextView	3
ImageView	4
Button	4
DrawRectangle	4
EditText	5
PARSING AND LOGIC	5
Samples	5
Publix Store	5
Hong's Buffet	8
UBIWIRELESS	9
FUTURE RELEASES AND FOCUS	9
THANKS	9

Introduction

The motivation to complete this application comes from the lack of an active navigation experience present on Google Maps or MapPoint services. The majority of information received in the map comes from a single source location and in general there is very limited capability for interaction user to establishment under the current model.

Google Android provides a flexible way to tackle location-based services, including MapActivities, MapViews, and Overlays. Ad2Location is a Google Android Application that provides you with an active map where a layer of points is combined with a layer of Images, Buttons, Texts, that are retrieved by a server and converted into a widget-type of interface in which end-users can interact, real-time with information or services provided by the establishment located at a determined "Latitude and Longitude"

APPLICATION INSTRUCTIONS

Ads and interfaces are provided as part of the map. Since there is no GPS available (real GPS) on the phone, the application has limited its scope to a few business located in Coral Springs, FL. The amount of services can then be easily increased as well as the context information may dynamically change anytime.

The app is then a GPS-like browser where each location in the map is depicted by a "BLUE" dot.. This Blue Dot can be Navigated thru by selecting MENU and then "Navigate to Ads" button.

INSTRUCTIONS

These instructions are displayed the first time you run the app, the app will show this only ONCE. Use "RIGHT And LEFT Cursor KEYS" to review the following instructions posted on the phone (Figure 1).

- Navigate thru the map using UP-DOWN keys ONLY.
- Once in a location/point, press the "FIRE KEY". (center button, also called Select The Ad is retrieved from our server and shown on the screen. You may navigate within the Ad as well as you did with the map.
- To go back to Map navigation you may press LEFT-RIGHT keys, the map will automatically move
- Once you are inside the Ad, you may select its items by using "UP-DOWN" keys, you may press "FIRE" key on top of images, buttons, or text fields
- Depending on how the XML from the particular "Vendor" is designed you may go thru multiple screens or a single one.

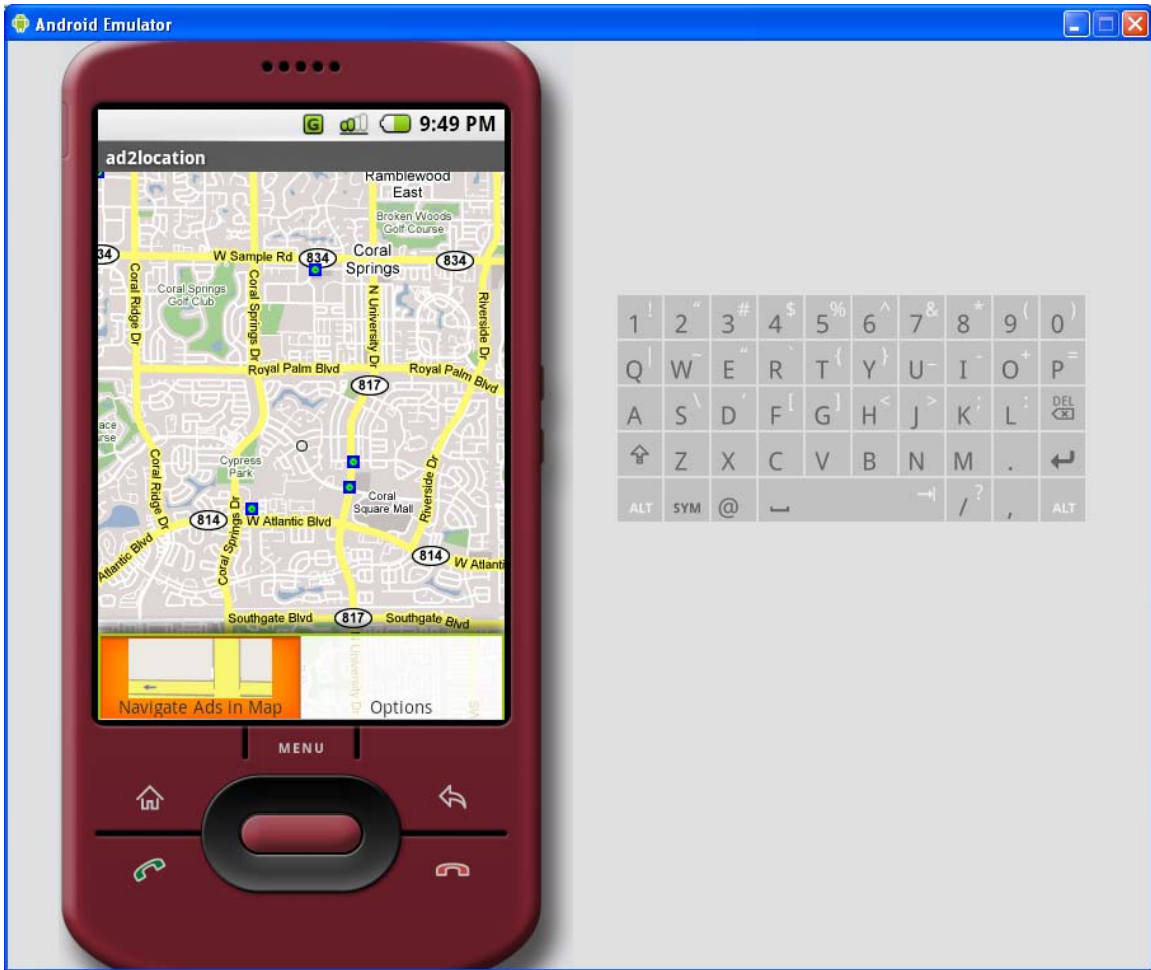


Figure 1. Simple Single-Button Navigation

Ad2Location Concept

Ad2Location is a mobile response to an effort to convert one of the layers of top of a map to be an interactive interface to the mobile user. The images shown in the map can then become your Canvas. The current model provides a mechanism in which you can find “Starbucks around Latitude or Longitude,” or find your best price available for X or Y around a zip code. However, when a user is already at certain vicinity, he may simply look around his local area, and from his cell phone and without leaving a map, may want to check, how many patrons are at local bar, what coupons the local supermarket may have available, or he/she may choose to make a reservation with a single click of a button.

Ad2Location provides the framework to complete these tasks, by:

- Enabling the map to be modifiable and accept user input when selecting an specific widget
- Enabling the vendor or service to include its own User Interface that is seen by any mobile user using Google Android
- Reuse of Google Android’s XML definition language with minor modifications to support Ad2Location

A service provider (e.g. Bookstore), may select to scan their paper ads, create a sequence of PNG or GIF files, and hire a freelance developer to use Android's XML schema to create a slideshow presenting its coupons. By downloading the ad into a site, the add will be clickable, as wlel as context information such as telephone numbers, email addresses, text messages, URL, can also be included as part of the advertisement.

For a vendor to create a new "Ad" or "Slide show" three steps are required:

- Ad2Location application installed with latest software supporting your XML schema
- Understanding this XML schema which is the same or similar to Google Android's definition.
- Post the service onto the site that can then be browsed by the mobile client.

Implementation Assumption

It's understood that you can enable filters for ads, such as "Search for Food," :or organized the Ads by "Distance." This application contains a core feature which is parsing the XML and managing a communication session thru the UI generated form an XML residing at a remote server location.

XML Definition Language

As follows, the XML language in use by Google Android is shown here. This definition language is similar to what is already in use by any Android App in any Layout. However a few changes were introduced, and none of the original Views were created on top of the Overlay.

In general four more fields are required per each entity:

- Android:x, which represents the X-coordinate where the item is going to be rendered on the screen
- Android:y, which represents the Y-coordinate
- Android:url, this indicates actually any URI value, at this point the application support intents for tel:, http:., and geo:
- Android:xmlurl, this is a new location where more XML will be downloaded if the user presses the "FIRE KEY" when selecting this particular element.

TextView

A sample TextView XML is show below. This XML uses x,y. From the representation bellow, you may find that Android:color could also be text or hexadecimal formal, also included android:alpha (0-255).

```
<TextView id="@+id/serif"
    android:text="(c) 2008 - ad2location"
    android:typeface="serif"
    android:x="55"
    android:y="180"
    android:color="RED">
</TextView>
```

ImageView

In this imageView, the source of the image is defined by android:src, but similar to TextView, android:x, android:y, and android.xmlurl, or android:url are in use here.

```
<ImageView id="@+id/button2"
    android.xmlurl="http://android.ubiwireless.com/publix.php"
    android:x="50"
    android:y="40"
    android:src="http://android.ubiwireless.com/ubiwireless.png"
    android:url="http://www.ubiwireless.com/">
</ImageView>
```

Button

A button is more complex, in reality that you can have text inside of it, but in general the same behavior occurs.

```
<Button id="@+id/button2"
    android:text="Call CEO"
    android:typeface="bold"
    android:url="tel:9547754153"
    android:width="90"
    android:height="30"
    android:x="60"
    android:y="200"
    android:color="Blue"
    android.colortext="Yellow">
</Button>
```

DrawRectangle

A DrawRectangle is general used for decoration or background to text or buttons in which you may draw on top of. Even though you may select or add Android.xmlurl and can select the image itse, its is not recommended go have any Action associated to it.

```
<DrawRectangle id="rectangle"
    android:x="40"
    android:y="30"
```

```
        android:color="WHITE"
        android:alpha="75"
        android:width="120"
        android:height="220">
</DrawRectangle>
```

EditBox

This is not yet fully supported, completion and basic editing of the field is left for a future release

Parsing and logic

A SAX parser is used to review the text a full XML. The XML needs to be complete and any invalid XML may lead the application to crash, however attributes may be added at will. Attributes not supported will not trigger an application failure, however missing some of the values (x, y) may lead the application to render everything on 0,0. A set of services and XML format can be retrieved from <http://android.ubiwireless.com/services.php>

A few samples are shown in the next section

Samples

Publix Store

The XML for the Publix Store is shown below, as you can see there are some textbox and a “Call Us button” (see FIG 2 a). FIG 2.a depicts the image created on the screen and rendered from this XML. The user may select “Call Us” in which the phone will initiate a call, or select “MORE” in which a new screen is shown as depicted in FIG 2.b

```
<DrawRectangle id ="@rectangle"
    android:x="100"
    android:y="100"
    android:color="GREEN"
    android:alpha="160"
    android:width="200"
    android:height="100">
</DrawRectangle>

<TextView xmlns:android="http://schemas.android.com/apk/res/android"
    android:text="PUBLIX"
    android:x="120"
    android:typeface="bold"
    android:size="16"
    android:y="120">
</TextView>
```

```

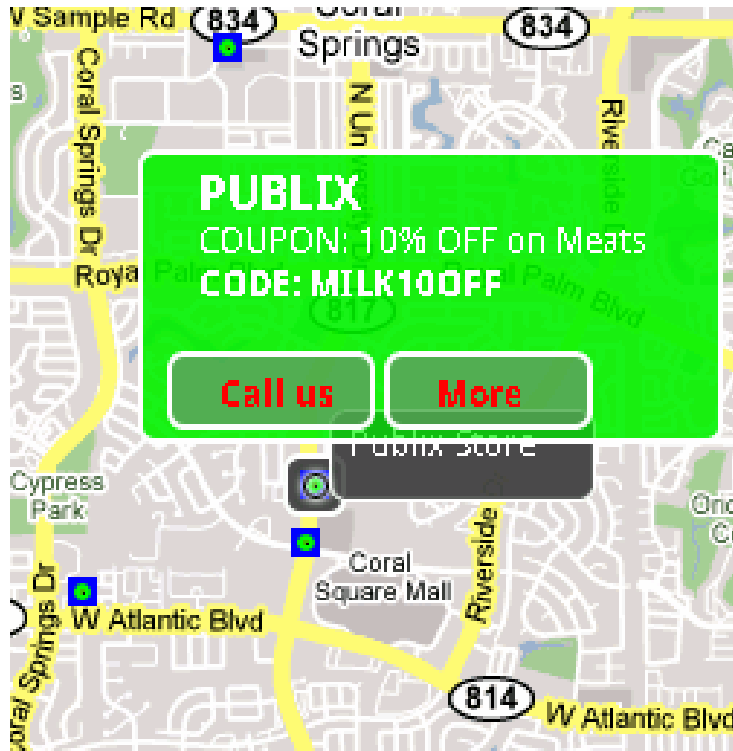
<TextView xmlns:android="http://schemas.android.com/apk/res/android"
    android:text="COUPON: 10% OFF on Meats"
    android:x="120"
    android:typeface="normal"
    android:size="12"
    android:y="135">
</TextView>

<TextView xmlns:android="http://schemas.android.com/apk/res/android"
    android:text="CODE: MILK10OFF "
    android:x="120"
    android:size="12"
    android:typeface="bold"
    android:y="150">
</TextView>

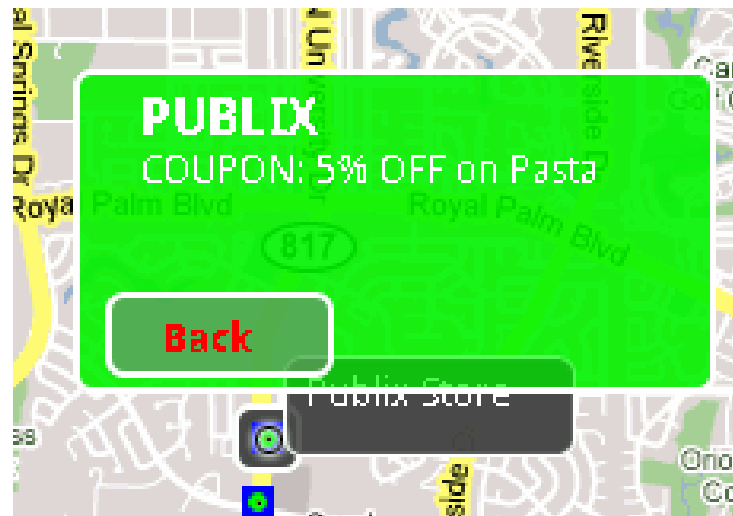
<Button xmlns:android="http://schemas.android.com/apk/res/android"
    android:text="Call us"
    android:x="110"
    android:color="Green"
    android:colortext="Red"
    android:typeface="bold"
    android:url="tel: 9547754155"
    android:y="170"
    android:width="70"
    android:height="25">
</Button>

<Button xmlns:android="http://schemas.android.com/apk/res/android"
    android:text="More"
    android:x="185"
    android:color="Green"
    android:colortext="Red"
    android:typeface="bold"
    android:xmlurl="http://android.ubiwireless.com/publix1.php"
    android:y="170"
    android:width="70"
    android:height="25">
</Button>

```



(a)



(b)

FIG 2.a. Screen Init from PUBLIX site, b) Folow up screen

To create FIG 2.b, the following XML is required.

```

DrawRectangle id="@rectangle"
  android:x="100"
  android:y="100"
  android:color="GREEN"
  
```

```

        android:alpha="160"
        android:width="200"
        android:height="100">
</DrawRectangle>

<TextView xmlns:android="http://schemas.android.com/apk/res/android"
    android:text="PUBLIX"
    android:x="120"
    android:typeface="bold"
    android:size="16"
    android:y="120">
</TextView>

<TextView xmlns:android="http://schemas.android.com/apk/res/android"
    android:text="COUPON: 5% OFF on Pasta"
    android:x="120"
    android:typeface="normal"
    android:size="12"
    android:y="135">
</TextView>

<Button xmlns:android="http://schemas.android.com/apk/res/android"
    android:text="Back"
    android:x="110"
    android:color="Green"
    android:colortext="Red"
    android:typeface="bold"
    android:xmlurl="http://android.ubiwireless.com/publix.php"
    android:y="170"
    android:width="70"
    android:height="25">
</Button>

```

Hong's Buffet

Hong's Buffet is a simpler concept, no buttons are created but a Android Logo is used as part of their “promotional campaign”

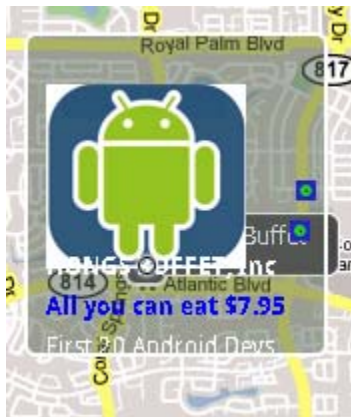


FIG 3. Hong's Buffet “special” for Google Android developers

UBIWIRELESS

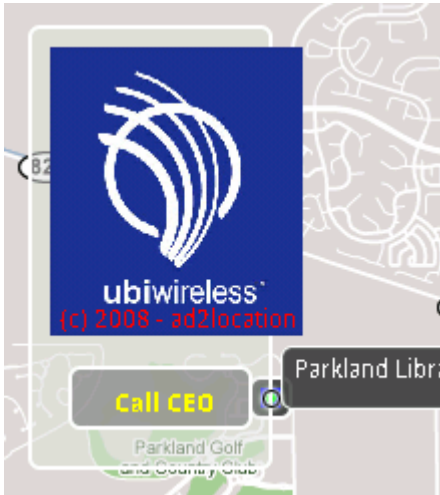


FIG 4. Ubiwireless UI

The majority of sites, will present a similar UI as this one.

Future Releases and Focus

Our team wants to focus on the following features for the next two weeks

- Short term
- EditText with completion, maybe use a subactivity for this
- Provisioning of applications and enabling `apk://` extensions.

Longer Term

- Speeding up Image downloading
- Fix UI controls
- Provide a WSYWYG interface to create ad2location definitions

THANKS

Thanks the guys on anddev.org which has provided a lot of guideline and understanding, specially overlay sample application posted on that site.