SKRIPSI

PENGEMBANGAN APLIKASI PENCARI RUTE KENDARAAN UMUM UNTUK WINDOWS PHONE

YOHAN

NPM: 2011730048

PROGRAM STUDI TEKNIK INFORMATIKA
FAKULTAS TEKNOLOGI INFORMASI DAN SAINS
UNIVERSITAS KATOLIK PARAHYANGAN
2014

UNDERGRADUATE THESIS

DEVELOPMENT APPLICATION PUBLIC TRANSPORT ROUTE SEARCH FOR WINDOWS PHONE

YOHAN

NPM: 2011730048

DEPARTMENT OF INFORMATICS
FACULTY OF INFORMATION TECHNOLOGY AND SCIENCES
PARAHYANGAN CATHOLIC UNIVERSITY
2014

LEMBAR PENGESAHAN

PENGEMBANGAN APLIKASI PENCARI RUTE KENDARAAN UMUM UNTUK WINDOWS PHONE

YOHAN

NPM: 2011730048

Bandung, 1 Juli 2014

Menyetujui,

Pembimbing Tunggal

Pascal Alfadian, M.Com.

Ketua Tim Penguji	Anggota Tim Penguji
Thomas Anung Basuki, Ph.D.	Dr. rer. nat. Cecilia Esti Nugraheni

Mengetahui,

Ketua Program Studi

Thomas Anung Basuki, Ph.D.

PERNYATAAN

Dengan ini saya yang bertandatangan di bawah ini menyatakan bahwa skripsi dengan judul:

PENGEMBANGAN APLIKASI PENCARI RUTE KENDARAAN UMUM UNTUK WINDOWS PHONE

adalah benar-benar karya saya sendiri, dan saya tidak melakukan penjiplakan atau pengutipan dengan cara-cara yang tidak sesuai dengan etika keilmuan yang berlaku dalam masyarakat keilmuan.

Atas pernyataan ini, saya siap menanggung segala risiko dan sanksi yang dijatuhkan kepada saya, apabila di kemudian hari ditemukan adanya pelanggaran terhadap etika keilmuan dalam karya saya, atau jika ada tuntutan formal atau non-formal dari pihak lain berkaitan dengan keaslian karya saya ini.

Dinyatakan di Bandung,
Tanggal 1 Juli 2014

Meterai

Yohan
NPM: 2011730048

ABSTRAK

Sedang Dalam Pembuatan.

Kata-kata kunci: Rute, Kendaraan Umum, Windows Phone

ABSTRACT

Under Construction.

Keywords: Route, Public Transport, Windows Phone

Dipersembahkan untuk diri sendiri

KATA PENGANTAR

Puji Syukur penulis kepada Tuhan yang telah memberikan rahmatnya sehingga penulis dapat menyelesaikan skripsi yang berjudul "Pencari Rute Kendaraan Untuk Windows Phone"

Bandung, Juli 2014

Penulis

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

BAB 1

PENDAHULUAN

Pada Bab satu akan dibahas pendahuluan dari penelitian yang dilakukan. Bab satu terbagi dalam delapan *sub bab*, yaitu *latar belakang*, *rumusan masalah*, *tujuan*, *batasan masalah*, *ruang lingkup masalah*, *metode penelitian*, *teknik pengumpulan data*, dan *sistematika penulisan.*

1.1 Latar Belakang

Transportasi menjadi bagian yang penting bagi manusia di saat penelitian ini dilakukan. Ada dua jenis transportasi bagi seseorang yaitu kendaraan umum dan kendaraan pribadi. Tapi sekarang ini banyak yang lebih memilih kendaraan pribadi dibanding kendaraan umum. Maraknya penggunaan kendaraan pribadi dan penambahan jalur kendaraan yang tidak sebanding banyaknya kendaraan menimbulkan kemacetan. Maraknya penggunaan kendaraan pribadi dikarenakan kurang nyamannya kendaraan umum dan kesulitan dalam menentukan kendaraan umum yang harus dinaiki. Banyaknya rute kendaran umum membuat orang kebingungan dalam memilih kendaraan umum menuju lokasi yang diinginkan. Seseorang cenderung malas untuk bertanya dan mencari rute yang efisien. Karena hal tersebut membuat seseorang lebih memilih menggunakan kendaraan pribadi ketimbang kendaraan umum.

Ide pembuatan aplikasi yang memudahkan seseorang dalam menentukan rute kendaraan umum sudah lebih dulu ada yang dikenal dengan nama Kiri. Kiri dibuat dengan latar belakang tiga masalah besar yaitu pemanasan global, kemacetan, dan harga bahan bakar minyak yang tinggi. Meskipun Kiri pertama dibuat di web tetapi Kiri dapat dimanfaatkan untuk pencarian kendaraan selain di web. Pemanfaatan Kiri tersebut dalam mencari rute kendaraan umum dengan mengggunakan Kiri API.

Pesatnya perkembangan teknologi sekarang ini mendorong perkembangan perangkat bergerak (*mobile*). Perangkat bergerak kian digemari orang-orang terutama di Indonesia. Salah satu yang menarik perhatian adalah Windows Phone 8 yang dibuat Microsoft. Antarmuka Windows Phone 8 yang disebut *Metro* cukup menarik dan mudah digunakan. Meskipun jumlah penggunanya masih belum sebanyak pengguna Android dan IOS tapi jumlah penggunanya terus naik di tahun 2014 ini.

Berdasarkan hal tersebut, penulis mencoba mengembangakan aplikasi Pencarian Rute Kendaraan Umum di Windows Phone dalam tugas akhir ini. Aplikasi yang penulis kembangan akan memungkinkan pengguna menemukan rute kendaraan umum untuk sampai di tujuan. Untuk memudahkan pengguna, penulis akan menampilkan dalam 2 bentuk yaitu peta dan daftar.

1.2 Rumusan Masalah

Sehubung dengan latar belakang diatas timbul permasalahan sebagai berikut:

- Bagaimana membuat aplikasi di Windows Phone?
- Bagaimana mengintegrasikan Kiri API dengan aplikasi pencari rute kendaraan umum di Windows Phone?

1.3 Tujuan

Berdasarkan rumusan masalah pada sub bab 1.2, maka tujuan dari pembuatan tugas akhir ini adalah:

- Mempelajari cara pembuatan perangkat linak di Windows Phone lalu mengembagan aplikasi yang akan dibuat.
- Membuat aplikasi di di Windows Phone yang memanfaatkan Kiri API.

1.4 Batasan Masalah

Ruang lingkup pengembangan aplikasi Pencari Rute Kendaraan untuk Windows Phone ini dibatasi hal berikut:

- Aplikasi ini akan berjalan di sistem operasi Windows Phone 8.
- Aplikasi ini membutuhkan koneksi internet.
- Aplikasi ini akan menampilkan rute jalur angkot, bus umum dan travel di tiga kota besar yaitu Bandung, Jakarta, dan Surabaya.

1.5 Ruang Lingkup Masalah

1.6 Metode Penelitian

Metode Penelitian yang penulis gunakan dalam membuat tugas akhir ini adalah sebagai berikut:

- Melakukan studi pustaka mengenai tombol di Windows Phone, navigation di Windows Phone, Map di Windows Phone, GPS di Windows Phone dan Kiri API.
- Melakukan analisis terhadap aplikasi lain yang menggunakan Kiri API.
- Melakukan analisis terhadap dasar teori untuk pembangunan aplikasi Pencarian Rute Kendaraan Umum untuk Windows Phone.
- Melakukan perancangan aplikasi Pencarian Rute Kendaraan Umum untuk Windows Phone.
- Implementasi dari aplikasi Pencarian Rute Kendaraan Umum untuk Windows Phone.

- Menguji aplikasi Pencarian Rute Kendaraan Umum untuk Windows Phone.
- Membuat kesimpulan.

1.7 Teknik Pengumpulan Data

1.8 Sistematika Penulisan

Bab 1 membahas latar belakang masalah, rumusan masalah, tujuan penulisan tugas akhir, batasan masalah, ruang lingkup masalah, metode penelitian, dan teknik pengumpulan data tugas akhir ini.

Bab 2 membahas tentang teori-teori yang akan digunakan dalam tugas akhir ini. Teori-teori yang dijelasakan mengenai Kiri API, *Web Service*, pembangunan aplikasi di Windows Phone, antarmuka di Windows Phone, dan algoritma yang dipakai.

Bab 3 membahas tentang analisi pembangunan aplikasi Pencarian Rute Kendaraan Umum untuk Windows Phone.

BAB 2

DASAR TEORI

Bab ini berisi dasar teori dari pembangunan Aplikasi Pencarian Rute Kendaraan Umum untuk Windows Phone. Beberapa teori yang dibahas dalam bab ini adalah Kiri API, Web Service, Menampilkan Peta, Penggunaan *Global Positioning System* di Windows Phone, dan antarmuka perangkat lunak yang dibuat.

2.1 Windows Phone

Sub bab ini akan membahas pemrograman di Windows Phone. Pembahasan akan dimulai dengan apa itu Windows Phone dan fitur di Windows Phone yang akan digunakan dalam pembangunan perangkat lunak Pencarian Rute Kendaraan di Windows Phone.

Windows Phone merupakan sistem operasi untuk perangkat bergerak yang dikembangkan Microsoft.[1] Untuk mengembangkan aplikasi Windows Phone dibutuhkan Windows Desktop 8 sebagai media pengembangan. Bahasa pemrograman yang digunakan untuk membuat perangkat lunak di Windows Phone yaitu C# dan Visual Basic.

2.1.1 Lingkungan Kerja

Microsoft .NET framework merupakan sebuah perangkat lunak yang dibangun untuk membantu dalam pembangunan aplikasi di Windows, Windows Phone, Windows Server, and Microsoft Azure[2]. Ini terdiri dari runtime bahasa umum dan perpustakaan kelas NET Framework, yang meliputi kelas, interface, dan jenis nilai yang mendukung berbagai teknologi. NET Framework menyediakan lingkungan yang mudah dikelola, pengembangan disederhanakan dan penyebaran, dan integrasi dengan berbagai bahasa pemrograman, termasuk Visual Basic dan Visual C#. Seperti yang telah disebutkan ada dua bahasa pemrograman dalam .NET Framework yang dipakai dalam pembangunan aplikasi di Windows Phone 8 yaitu Visual Basic dan Visual C#. Untuk masalah kehandalan keduanya menawarkan kehandalan yang baik. Kelebihan dari Visual Basic adalah bahasa pemrograman berorientasi objek yang kuat dan memiliki banyak pengenbangan fitur di inheritance, polymorphism, interfaces, and overloading[3].Kelebihan dari C# yang merupakan pengembangan dari C/C++ adalah sederhana, modern, aman dan berorientasi objek[4]. Satu hal yang dirasakan penulis adalah kenyamanan ketika memilih bahasa .NET tersebut. Akan lebih mudah bagi developer yang menggunakan Visual Basic 6.0 untuk menggunakan Visual Basic .NET. Tetapi bagi deveoper

[1] en.wikipedia.org/wiki/Windows_Phone

yang menggunakan C++ atau java sebelumnya akan lebih mudah menggunakan C#.

2.1.2 XAML

Extensible Application Markup Language (XAML) merupakan bahasa deklaratif yang dipakai untuk membuat antarmuka aplikasi. XAML merupakan bahasa yang digunakan untuk membuat antarmuka di Windows Phone 8. Pada dasarnya penggunaan XAML sama dengan HTML pada pembuatan antarmuka web. XAML dapat menginisialisasi objek dan mengatur properti untuk menunjukan hubungan antar objek. Untuk aturan penulisan sintak XAML didasarkan pada XML. Setiap XAML Windows Runtime menggunakan konvensi XAML language dan ditulis pada *namespace* yang ditandai dengan prefix x sebagai elemen paling atas. Setelah itu di baris ke dua dimulai dengan xmlns diikuti titik dua, lalu nama dari *namespace*, diikuti tanda sama dengan dan *path* perepresentasian *namespace*. Prefix x pada XAML mengandung beberapa struktur program yang sering kita gunakan yaitu :

- x:Key : sebuah nama unik untuk menunjuk referensi ke suatu resource atau berkas lain. Nilai ini dapat dipanggil kembali untuk menggunakan resource tersebut.
- x:Class : menunjukkan nama kelas.
- x:Name : menunjukkan nama sebuah obyek dan untuk membedakan antar obyek yang satu dengan obyek yang lain.
- x:Uid : mengidentifikasi elemen objek dalam XAML. Objek elemen merupakan objek yang dapat melakukan kontrol terhadap kelas atau elemen lain yang ditampilkan di desain anarmuka.

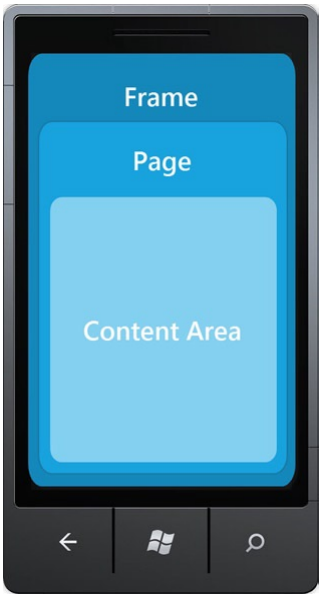
2.1.3 Kontrol terhadap Ponsel

Kontrol terhadap ponsel yang dimaksudkan disini adalah pengaturan tata letak terhadap antarmuka di Windows Phone. Windows Phone 8 menyediakan banyak set kontrol yaitu tata letak, tombol, kontrol masukan untuk mendapatkan informasi sampai ke menu.

2.1.3.1 Navigasi

Aplikasi yang dibuat di Windows Phone didasarkan pada model halaman. Maksud dari model halaman adalah pengguna berpindah dari satu halaman ke halaman lain dengan konten yang berbeda-beda dengan frame sebagai pengontrolnya. Setiap antarmuka aplikasi dibungkus dengan frame. Frame inilah yang melakukan kontrol terhadap aplikasi dan memungkinkan berpindah dari satu halaman ke halaman lain. Sedangkan halaman merupakan pembungkus dari elemen di dalamnya saja. Untuk lebih jelas mengenai frame, halaman dan area konten dapat dilihat pada gambar di bawah ini.

[2] http://msdn.microsoft.com/en-us/library/vstudio/w0x726c2%28v=vs.110%29
[3] http://msdn.microsoft.com/en-us/library/aa903378%28v=vs.71%29.aspx
[4] http://msdn.microsoft.com/en-us/library/aa287558%28v=vs.71%29.aspx



Gambar 2.1: Hirarki Navigasi

2.1.3.2 Kontrol Tata Letak

Kontrol Tata Letak merupakan penampung pada antarmuka Windows Phone untuk objek di antarmuka dan kontrol yang lain (tombol radio, textbox, dan lai-lain). Kontrol tata letak digunakan untuk meletakan objek-objek di layar. Ketika pertama membuat aplikasi Windows Phone maka tata letak dasar sebagai penampung akan langsung dibuat berikut panel judul dan panel konten. Selanjutnya untuk penambahan kontrol tata letak yang lain dapat ditambahkan di panel konten.

Ada 3 macam panel yang dipakai untuk menangani Tata Letak yaitu Grid, StackPane, dan Canvas. Perlu diperhatikan bahwa setiap halaman hanya memiliki satu macam panel. Berikut 3 macam panel di Windows Phone:

- StackPanel merupakan panel yang memposisikan element menjadi 1 baris dan beberapa elemen di setiap halaman diposisikan horizontal atau vertical saja.
- Grid merupakan panel yang mendukung tata letak yang rumit. Panel ini memposisikan elemen di baris dan kolom mana saja di setiap halaman.
- Canvas memposiskan elemen sebagai absolut kordinat. Jadi setiap elemen di dalam Canvas dapat diposisikan spesifik sesuai kordinat x dan y.

2.1.3.3 Kontrol Terhadap Teks

Kontrol Terhadap Teks secara menampilkan konten String. Ada berbagai macam Kontrol Terhadap Teks di Windows Phone yaitu TextBlock, TextBox dan PasswordBox. Ketiga macam kontrol tersebut dibedakan menurut tujuannya. Berikut gambar dan keterangan masing-masing:

- TextBlock merupakan tempat menaruh potongan teks yang hanya bisa dilihat.
- TextBox biasanya digunakan untuk teks masukan yang pendek. Tapi bisa juga dipakai untuk masukan yang banyak dan beberapa baris.

Gambar 2.2: TextBlock, TextBox dan PasswordBox

- PasswordBox biasanya digunakan untuk masukan yang bersifat rahasia. Karakter yang dimasukan langsung disamarkan menjadi bentuk titik.

2.1.3.4 Tombol dan Kontrol Pilihan

Tombol memungkinkan pengguna untuk bernavigasi. Sedangkan kontrol pilihan memudahkan dalam memilih. Berikut gambar dan keterangan masing-masing:

- Button merupakan kontrol yang dipakai pengguna untuk mengaktifkan *event* klik.
- HyperlinkButton merupakan kontrol yang menampilkan hyperlink. Jika di tekan maka akan menunjuk ke halaman yang dituju.
- CheckBox merupakan kontrol yang memungkinkan pengguna memilih beberapa item.
- RadioButton merupakan kontrol yang memungkinkan pengguna memilih satu pilihan dari beberapa pilihan.
- Slider merupakan kontrol yang memungkinkan user memilih nilai kisaran dari jalur yang sudah disediakan.

2.1.3.5 Kontrol Daftar

Kontrol yang dipakai untuk menampilkan daftar dari beberapa item. Berikut keterangan masing-masing:

- ListBox akan menampilkan daftar item. Daftar ini dapat dipilih dengan cara di klik.

Gambar 2.3: TextBlock, TextBox dan PasswordBox

- LongListSelector dipakai untuk mengelompokan, menampilkan, dan melakukan penggulungan terhadap daftar yang panjang.

2.1.3.6 Gambar, Peta, dan Media

Menampilkan gambar, map, dan konten media penting sebagai bagian dari antarmuka. Berikut keterangan masing-masing:

- Image dipakai untuk menampilkan gambar. Aplikasi di Windows Phone mendukung format jpeg dan png.
- Map dipakai menampilkan peta.
- MediaElement dipakai untuk memainkan audio dan video.

2.1.4 Lifecycle Windows Phone

- Running : Aplikasi sedang berada di bagian depan. Hanya satu aplikasi yang diijinkan berada di depan pada satu waktu.
- Dormant : Aplikasi tidak berada di bagian depan, tetapi sistem operasi tidak membutuhkan sumber daya(dalam hal ini memori). Pada tahap ini objek aplikasi masih berada di memori, jadi pada tahap Dormant aplikasi dapat secara otomatis diaktifkan kembali. Tetapi jika sistem operasi membutuhkan sumbe daya maka aplikasi akan menjadi *tombstoned* dan dihancurkan.
- Tombstoned : Aplikasi akan dihancurkan dan objek aplikasi akan dihapus dari memori. Ketika aplikasi diaktivasi pengguna akan ditempatkan kembali di halaman yang ditinggalkan sebelumnya, tetapi objek harus di kembalikan dari keadaan sebelumnya.

2.1.4.1 Penyimpanan dan Pengembalian Keadaan Aplikasi

Listing 2.1: App.xaml

```
<Application.ApplicationLifetimeObjects>
    <!--Required object that handles lifetime events for the application-->
    <shell:PhoneApplicationService
        Launching="Application_Launching" Closing="Application_Closing"
        Activated="Application_Activated" Deactivated="Application_Deactivated"/>
</Application.ApplicationLifetimeObjects>
```

Listing 2.2: Appp.xaml.cs

```
// Code to execute when the application is launching (eg, from Start)
// This code will not execute when the application is reactivated
private void Application_Launching(object sender, LaunchingEventArgs e)
{
}
// Code to execute when the application is activated (brought to foreground)
// This code will not execute when the application is first launched
private void Application_Activated(object sender, ActivatedEventArgs e)
{
}
// Code to execute when the application is deactivated (sent to background)
// This code will not execute when the application is closing
private void Application_Deactivated(object sender, DeactivatedEventArgs e)
{
}
// Code to execute when the application is closing (eg, user hit Back)
// This code will not execute when the application is deactivated
private void Application_Closing(object sender, ClosingEventArgs e)
{
}
```

Listing 2.3: Penanganan terhadap pengaktifan dan penonaktifan kejadian

```
public class MyObject
{
public DateTime LastUpdate { get; set; }
}
//...
public MyObject MyObject { get; set; }

private void Application_Activated(object sender, ActivatedEventArgs e)
{
    DateTime lastUpdate =
    (DateTime)PhoneApplicationService.Current.State["LastUpdate"];
    Debug.WriteLine("Application_Activated: LastUpdate=" + lastUpdate.ToLongTimeString());
}
private void Application_Deactivated(object sender, DeactivatedEventArgs e)
{
    Debug.WriteLine("Application_Deactivated: Saving LastUpdate to application state");
    PhoneApplicationService.Current.State["LastUpdate"] = DateTime.Now;
}
```

2.1.5 Peta di Windows Phone

Peta untuk Windows Phone adalah Here Maps. Here Maps yang akan menampilkan lokasi dan melakukan penelusuran. Pada Sub Bab ini akan dibahas mengenai penambahan peta pada aplikasi, mendapatkan lokasi, petunjuk arah dan Pushpins.

2.1.5.1 Penambahan Maps Ke Aplikasi

Ada dua cara untuk menambahkan peta ke aplikasi yang pertama dengan menjalankan MapTask dan yang kedua dengan ppenambahan ID_CAP_MAP ke \properties\WMAppManifest.xml. Contoh dibawah adalah dengan MapTask yang memenggil metode Show() untuk memunculkan peta.

Listing 2.4: code pemanggilan maps

```
private void MapClick(object sender, EventArgs e)
```

```
2     {
3         var task = new MapsTask();
4         var goldenGateBridge = new GeoCoordinate(37.8085880, -122.4770175);
5         task.Center = goldenGateBridge;
6         task.ZoomLevel = 9;
7         task.SearchTerm = "Golden Gate Bridge";
8         task.Show();
9     }
```

Listing 2.5: penambahan Map Element menggunakan Map Control

```
1     <Grid x:Name="LayoutRoot" Background="Transparent">
2         <Controls:Map x:Name="WorldMap" />
3     </Grid>
```

2.1.5.2 Posisi Pada Peta

Ketika memulai membuka peta maka akan menampilkan seluruh dunia dan pengguna harus melakukan *pinch-and-zoom* untuk melihat titk tertentu. Agar dapat memudahkan pengguna dalam memilih titik tertentu secara otomatis maka dapat menggunakan metode SetView. Pada Listing dibawah dapat dilihat bahwa SetView memiliki 2 parameter. Parameter pertama menyatakan tempat dengan GeoCordinate yang mendefinisikan pusat pandangan. Selanjutnya parameter ke 2 merupakan zoomLevel (1 = zoomed out, 20 = zoomed in).

Listing 2.6: Posisi Pada Peta

```
1     GeoCoordinate GoldenGateBridge = new GeoCoordinate(37.8085880, -122.4770175);
2     SanFranciscoMap.SetView(GoldenGateBridge, 15);
```

2.1.5.3 Penambahan Pushpins ke Peta

Peta tidak mendukung langsung cara untuk mengikat data ke MapLayer dan MapOverlay. Untungnya di Windows Phone memiliki Windows Phone 8 Toolkit yang memiliki set objek yang dapat digunakan sebagai jembatan mengikat data ke MapLayer dan MapOverlay.

Listing 2.7: Berikut contoh Tookkit Pushpin dengan elemen-element yang di tulisa manual

```
1     <Controls:Map>
2         <Toolkit:MapExtensions.Children>
3             <Toolkit:Pushpin Content="London" GeoCoordinate="51.499493,-0.124753" />
4         </Toolkit:MapExtensions.Children>
5     </Controls:Map>
```

Gambar dibawah merupakan keluaran di peta. Label "London" merupkan bawaan Windows Phone 8 pushpin.

Selain menetapkan Pushpin secara langsung di MapExtensions, dapat pula diletakan MapItemsControl. Pushpin akan berpindah ke ItemTemplate dan ItemSource akan ditugaskan ViewModel.

Listing 2.8: Berikut contoh Tookkit Pushpin dengan elemen-element yang di tulisa manual

```
1     <Controls:Map x:Name="WorldMap" >
2         <Toolkit:MapExtensions.Children>
3             <Toolkit:MapItemsControl ItemTemplate="{StaticResource MapItemTemplate}" ItemsSource="{Binding
                  Source={StaticResource LocationViewModel}, Path=Locations}" />
4         </Toolkit:MapExtensions.Children>
5     </Controls:Map>
```

Listing 2.9: View Model

```
1     public class Location
2     {
```

Gambar 2.4: Keluaran Toolkit Pushpin pada Peta

```
3           public string Title { get; set; }
4           [TypeConverter(typeof(GeoCoordinateConverter))]
5           public GeoCoordinate GeoCoordinate { get; set; }
6       }
7
8       public class LocationViewModel
9       {
10          public ObservableCollection<Location> Locations { get; set; }
11          public LocationViewModel()
12          {
13              Locations = new ObservableCollection<Location>();
14          }
15      }
```

Listing 2.10: Deklarasi View Model

```
1       <phone:PhoneApplicationPage.Resources>
2           <vm:LocationViewModel x:Key="LocationViewModel" >
3               <vm:LocationViewModel.Locations>
4                   <vm:Location Title="London" GeoCoordinate="51.499493,-0.124753" />
5                   <vm:Location Title="Paris" GeoCoordinate="48.858222,2.2945" />
6                   <vm:Location Title="Rome" GeoCoordinate="41.890268,12.492315" />
7               </vm:LocationViewModel.Locations>
8           </vm:LocationViewModel>
9           <DataTemplate x:Name="MapItemTemplate">
10              <Toolkit:Pushpin GeoCoordinate="{Binding GeoCoordinate}" Content="{Binding Title}" />
11          </DataTemplate>
12      </phone:PhoneApplicationPage.Resources>
```

2.1.5.4 Mendapatkan Posisi Pengguna

Di Windows Phone 8 telah ada GeoCoordinate yang dapat digunakan untuk mengetahui posisi pengguna. Geolocator dari Windows.Devices.Geolocation akan mengembalikan posisi saat ini. Tetapi untuk menggunakan Geolocator, penulis perlu menghidupkan ID_CAP_LOCATION di \properties\WMAppManifest.xml.

Contoh kode dari penggunaan menggunakan XAML didefinisikan pada listing dibawah. UserLocationMarker di MapExtensions Children.UserLocationMarker merupakan Toolkit objek Windows Phone 8. Visibilitas UserLocationMarker di set "collapsed" agar penanda tetap berada di tempat sampai ada perubahan lokasi dan siap ditampilkan.

Listing 2.11: Map dan UserLocationMarker XAML

```
1       Grid x:Name="ContentPanel" Grid.Row="1" Margin="12,0,12,0">
2           <Controls:Map x:Name="WorldMap">
3               <Toolkit:MapExtensions.Children>
4                   <Toolkit:UserLocationMarker x:Name="locationMarker" Visibility="Collapsed" />
5               </Toolkit:MapExtensions.Children>
6           </Controls:Map>
7       </Grid>
```

Listing Dibawah akan mendemokan bagaimana GeoLocator mendapatkan posisi terkini. Contoh dari metode menggunakan *event* click. setelah itu dipanggil GetGeopositionAsync() yang akan mengembalikan objek GeoPosition object.GetGeopositionAsync yang sip menunggu, jadi tambahkan async untuk pemanggilan metode dan kata kunci await untuk mendapatkan nilai kembalian. Geoposition mengandung kordinat yang tidak kompatibel dengan GeoCoordinate() yang dipakai untuk map. Untungnya toolkit menyediakan metode ToGeoCoordinate() untuk menerjemahkan ke GeoCoordinate secara otomatis. Kode dibawah juga menjadi referensi pemakaian UserLocationMarker, makes it visible, dan di set menjadi GeoCordinate baru. Terakhir, metode Map SetView() dipanggil untuk menjadi pusat dari posisi yang baru.

Listing 2.12: Setting ulang Geolocator

```
1   private Geolocator _locator;
2   protected override void OnNavigatedTo(NavigationEventArgs e)
3   {
4       base.OnNavigatedTo(e);
5       _locator = new Geolocator()
6       {
7           // set either ReportInterval (milliseconds) or MovementThreshold (meters)
8           ReportInterval = 5000,
9           //DesiredAccuracy = PositionAccuracy.High
10          DesiredAccuracyInMeters = 1
11      };
12      _locator.PositionChanged += _locator_PositionChanged;
13  }
```

Listing 2.13: Melepas Geolocator yang dapat dipakai untuk melindungi sumber daya

```
1   protected override void OnNavigatedFrom(NavigationEventArgs e)
2   {
3       _locator.PositionChanged -= _locator_PositionChanged;
4       _locator = null;
5       base.OnNavigatedFrom(e);
6   }
```

Listing 2.14: Menangani Perubahan Posisi

```
1   void _locator_PositionChanged(Geolocator sender, PositionChangedEventArgs args)
2   {
3       this.Dispatcher.BeginInvoke(() =>
4           {
5               var geoCoordinate = args.Position.Coordinate.ToGeoCoordinate();
6               var locationMarker = this.FindName("locationMarker") as UserLocationMarker;
7               if (locationMarker != null)
8               {
9               locationMarker.Visibility = System.Windows.Visibility.Visible;
10              locationMarker.GeoCoordinate = geoCoordinate;
11              WorldMap.SetView(geoCoordinate, 15);
12              }
13          });
14  }
```

Listing 2.15: Menangani Perubahan Status

```
1   void _locator_StatusChanged(Geolocator sender, StatusChangedEventArgs args)
2   {
3       this.Dispatcher.BeginInvoke(() =>
4       {
5           var textBlock = this.FindName("StatusText") as TextBlock;
6           textBlock.Text = "Status: " + args.Status.ToString();
7       });
8   }
```

Berikut nilai yang mungkin dari Status Posisi:

- *Ready* : Jika lokasi tersedia.
- *Initializing* : Status jika penangkap GPS belum memiliki cukup satelit untuk mendapatkan posisi yang akurat.

- *NoData* : Data lokasi belum tersedia. Status ini muncul jika aplikasi sedang mamanggil GetGeopositionAsync atau register.
- *Disable* : Status mengindikasikan tidak diperbolehkannya pengaksesan lokasi.
- *NotInitialized* : Data lokasi belum tersedia. Status ini muncul jika aplikasi belum mamanggil GetGeopositionAsync atau register.
- *NotAvailable* : Jika Windows sensor dan lokasi tidak tersedia.

2.1.5.5 Arah

Penambahan langkah demi langkah arah rute di aplikasi di Windows Phone 8 mudah dilakukan. Untuk memunculkan arah cukup memiliki titik awal dan titik akhir. Pada listing dibawah mendemontrasikan pengarahan di map antara 2 titik. Untuk titik awal dan akhir perlu label lokasi dengan objek LabeledMapLocation. Selanjutnya setelah menetapkan dua objek pada MapsDirectionsTask lalu panggil metode show() dan arah dari titk awal ke titik akhir akan muncul di peta.

Listing 2.16: Penggunaan Arah pada Map

```
1  var wharf = new LabeledMapLocation()
2  {
3      Label = "Wharf",
4      Location = new GeoCoordinate(36.96252, -122.023372)
5  };
6  var park = new LabeledMapLocation()
7  {
8      Label = "Lighthouse Park",
9      Location = new GeoCoordinate(36.95172, -122.026783)
10 };
11 var task = new MapsDirectionsTask() { Start = wharf, End = park };
12 task.Show();
```

Meskipun Map bisa menggambar rute secara otomatis ada satu cara lagi untuk menambahkan objek tute pada Map yaitu menggunakan RouteQuery. Pertama yang harus didefinisikan pada RouteQuery yaitu GeoCoordinate yang harus dikunjungi. Pada listing dibawah dapat dilihat 3 GeoCoordinate dijadikan Waypoints di RouteQuery yang baru. RouteOptimization di pakai untuk meminimalisir jarak dan TraveMode di setel Driving.

Listing 2.17: Mengatur Ulang Rute Query

```
1  var wharf = new GeoCoordinate(36.96252, -122.023372);
2  var park = new GeoCoordinate(36.95172, -122.026783);
3  var lagoon = new GeoCoordinate(36.963365, -122.031932);
4  var routeQuery = new RouteQuery()
5  {
6      Waypoints = new List<GeoCoordinate>() { wharf, park, lagoon },
7      RouteOptimization = RouteOptimization.MinimizeDistance,
8      TravelMode = TravelMode.Driving
9  };
10 routeQuery.QueryCompleted += routeQuery_QueryCompleted;
11 routeQuery.QueryAsync();
```

Untuk melakukan handle terhadap QueryCompleted, pertama harus di periksa apakah terdapat kesalahan. Kembaliannya adalah objek rute. Lalu jadikan MapRoute konstruktor. MapRoute menambahkan visualisasi garis pada peta. Contoh implementasinya dapat dilihat di listing dibawah.

Listing 2.18: Melakukan Handle Terhadap QueryCompleted

```
1  void routeQuery_QueryCompleted(object sender, QueryCompletedEventArgs<Route> e)
2  {
3      if (e.Error == null)
4      {
```

```
5               this.WorldMap.AddRoute(new MapRoute( e.Result));
6               this.WorldMap.SetView(e.Result.BoundingBox);
7           }
8       }
```

Listing 2.19: Melakukan Perubahan Warna Rute di Map

```
1       var mapRoute = new MapRoute(e.Result)
2       {
3           Color = Colors.Red,
4           RouteViewKind = RouteViewKind.UserDefined
5       };
6       this.WorldMap.AddRoute(mapRoute)
```

Listing 2.20: Menampilkan Arah dalam Bentuk List

```
1       var sb = new StringBuilder();
2       var i = 0;
3       sb.AppendFormat("Estimated time: {0} minutes\n",
4           e.Result.EstimatedDuration.TotalMinutes.ToString());
5       foreach (var leg in e.Result.Legs)
6       {
7           foreach (var maneuver in leg.Maneuvers)
8           {
9               sb.AppendFormat("{0}. {1}: {2}\n",++i, maneuver.InstructionKind.ToString(), maneuver.
                    InstructionText);
10          }
11      }
12      MessageBox.Show(sb.ToString());
```

2.1.6 Memanfaatkan Sumber Data

2.1.6.1 Serializing Object

Serialization disini merupakan proses mentransformasikan objek ke format yang bisa dengan mudah dikirim melewati jaringan atau disimpan di database. Formatnya disini berupa string yang direpresentasikan sebagai objek di XML atau JSON(Javascript Object Notation). Ada beberapa objek yang dapat melakukan serialisasi, tetapi yang akan dibahas penulis disini hanya serialisasi JSON.

Banyak *web service* yang mengembalikan data dalam format JSON. JSON memiliki struktur yang mudah dipahami dimana kurung kurawal mengindikasikan objek, kurung siku berarti array, dan properti berupa nama dan nilai pasangan yang dipisahkan oleh titik dua. JSON format memiliki ukuran data yang kecil dan baik untuk penggunaan perangkat bergerak. Untuk contoh format JSON dapat dilihat di bagian Kiri API pada Bab 2 ini karena Kiri API menggunakan format JSON. Serialisasi menggunakan DataContractJsonSerializer membuat serialisasi mudah untuk menerjemahkan form String JSON ke objek yang dapat langsung digunakan. DataContractJsonSerializer memakai WriteObject() untuk serialisasi and ReadObject() untuk de-serialisasi.

2.1.6.2 Memanfaatkan Sumber Data Web

Listing 2.21: Pembentuk getPublicPhotos Url pada flicker

```
1       private const string BaseUrl = "http://ycpi.api.flickr.com/services/rest/";
2       private const string QueryStrings =
3       "?method={0}&api_key={1}&user_id={2}&format=json&nojsoncallback=1";
4       private const string FlickrMethod = "flickr.people.getPublicPhotos";
5       private const string YourApiKey = "<replace with api key here>";
6       private const string LibraryOfCongressKey = "8623220@N02";
7       private string FlickrPhotosUrl = BaseUrl +
8       String.Format(QueryStrings, FlickrMethod, YourApiKey, LibraryOfCongressKey);
```

Listing 2.22: Contoh JSON String dari flickr API

```
{
    "photos":{
        "page":1,
        "pages":193,
        "perpage":100,
        "total":"19222",
        "photo":[
            {
                "id":"9319237625",
                "owner":"8623220@N02",
                "secret":"e0d73d680b",
                "server":"5537",
                "farm":6,. . .
```

Selanjutnya untuk menerjemahkan JSON ke C# bisa lewat web sites json2csharp.com. Menuju ke json2csharp.com lalu paste JSON di text box yang ada dan click Generate. Selanjutnya buat class berekstensi .cs lalu paste text C# yang telah di generate.

Listing 2.23: Contoh JSON String dari flickr API

```
using System.Collections.Generic;

namespace ConsumingXML.Classes
{
    public class Photo
    {
        public string id { get; set; }
        public string owner { get; set; }
        public string secret { get; set; }
        public string server { get; set; }
        public int farm { get; set; }
        public string title { get; set; }
        public int ispublic { get; set; }
        public int isfriend { get; set; }
        public int isfamily { get; set; }
    }
    public class Photos
    {
        public int page { get; set; }
        public int pages { get; set; }
        public int perpage { get; set; }
        public string total { get; set; }
        public List<Photo> photo { get; set; }
    }
    public class RootObject
    {
        public Photos photos { get; set; }
        public string stat { get; set; }
    }
}
```

Listing 2.24: Mengembalikan daftar FlickrPhoto

```
private static List<FlickrPhoto> GetFlickrPhotos(string json)
{
const string baseUrl =
"http://farm{0}.staticflickr.com/{1}/{2}_{3}_s.jpg";
List<FlickrPhoto> FlickrPhotos = null;
var serializer = new DataContractJsonSerializer(typeof(RootObject));
using (var stream = new MemoryStream(Encoding.UTF8.GetBytes(json)))
{
var root = serializer.ReadObject(stream) as RootObject;
FlickrPhotos = (from photo in root.photos.photo
select new FlickrPhoto
{
Title = photo.title,
Uri = new Uri(String.Format(baseUrl,
photo.farm, photo.server, photo.id, photo.secret))
}).ToList();
}
return FlickrPhotos;
}
```

Listing 2.25: Menggunakan WebClient untuk mendownload String

```
private void UseWebClient()
```

```
2       {
3       var uri = new Uri(FlickrPhotosUrl);
4       var client = new WebClient();
5       client.DownloadStringCompleted += (sender, e) =>
6       {
7       var photos = GetFlickrPhotos(e.Result);
8       Dispatcher.BeginInvoke(() =>
9       {
10      FlickrListBox.DataContext = photos;
11      });
12      };
13      client.DownloadStringAsync(uri);
14  }
```

Listing 2.26: XAML untuk halaman

```
1       <Grid x:Name="LayoutRoot" >
2           <ListBox x:Name="FlickrListBox" ItemsSource="{Binding}">
3               <ListBox.ItemTemplate>
4                   <DataTemplate>
5                       <StackPanel Orientation="Horizontal">
6                           <Image Stretch="UniformToFill" Width="100" Height="100" Source="{Binding Uri}" />
7                           <TextBlock Grid.Column="1" Margin="10,0,0,0" Text="{Binding Title}"
                                HorizontalAlignment="Stretch" />
8                       </StackPanel>
9                   </DataTemplate>
10              </ListBox.ItemTemplate>
11          </ListBox>
12      </Grid>
```

2.2 Kiri API

Sub bab ini akan membahas Dokumentasi dari Kiri API. Pembahasan dimulai dengan pengantar dari Kiri API dan *Web Service*.

2.2.1 Pengantar Kiri API

API atau *Application Programming Interface* merupakan aturan yang dikodekan secara spesifik yang dapat digunakan untuk komunikasi antar aplikasi. Jadi API disini memfasilitasi untuk pemanggilan fungsi-fungsi tertentu diluar aplikasi itu sendiri. Pemanfaatan Kiri API adalah JSON atau *JavaScript Object Notation* format. Pemanfaatan Kiri API cukup dengan melakuan *request* dengan parameter dan Kiri akan mengembalikan hasil dalam format JSON. Untuk setiap *request* membutuhkan *API key* yang didapat dengan mendaftar[2]. Penggunaan API memungkinkan pengaksesan dimana saja dengan menggunakan koneksi internet. Pada Sub Bab dibawah penulis akan membahas beberapa pemanggilan pesan yang terdapat pada Kiri API.

2.2.2 Routing Web Service

Routing Web Service merupakan Kiri API yang digunakan untuk mendapatkan langkah perjalanan dari lokasi asal ke lokasi tujuan.

Berikut parameter *request* yang diperlukan berikut penjelasanya:

[2] https://bitbucket.org/projectkiri/kiri_api/wiki/KIRIAPIv2Documentation

version	2	Memberitahukan bahwa layanan yang dipakai adalah protokol veris 2
mode	"findroute"	mengintruksikan layanan untuk mencari rute
locale	"en" or "id"	bahasa yang digunakan untuk balasan
start	lat,lng (both are decimal values)	titk awal *Latitude* dan *longitude*
finish	lat,lng (both are decimal values)	titik akhir *Latitude* dan *longitude*
presentation	"mobile" or "desktop"	Menentukan tipe prensentasi untuk keluaran. Contoh, jika tipe presentasi "mobile", maka link "tel:" akan ditambahkan di hasil.
apikey	16-digit hexadecimals	API key yang digunakan

Berikut format Kiri API *responds*:

Listing 2.27: code *respond* pencarian rute

```
1  {
2      "status": "ok" or "error"
3      "routingresults": [
4          {
5              "steps": [
6                  [
7                      "walk" or "none" or others,
8                      "walk" or vehicle_id or "none",
9                      ["lat_1,lon_1", "lan_2,lon_2", ... "lat_n,lon_n"],
10                     "human readable description, dependant on locale",
11                     URL for ticket booking or null (future)
12                 ],
13                 [
14                     "walk" or "none" or others,
15                     "walk" or vehicle_id or "none",
16                     ["lat_1,lon_1", "lan_2,lon_2", ... "lat_n,lon_n"],
17                     "human readable description, dependant on locale",
18                     URL for ticket booking or null (future)
19                 ]
20             ],
21             "traveltime": any text string, null if and only if route is not found.
22         } ,
23         {
24             "steps": [ ... ],
25             "traveltime": "..."
26         } ,
27         {
28             "steps": [ ... ],
29             "traveltime": "..."
30         } ,
31         ...
32     ]
33 }
```

Berikut maksud dari listing 2.1: Ketika pencarian rute sukses dilakukan maka status akan memberitahukan "ok" seperti di baris 2. Selanjutnya setiap langkah dari posisi awal ke posisi tujuan akan ditampung di array dari langkah. Berikut keterangan dari setiap array tersebut:

- Index ke 0 atau baris 7 pada listing 2.1 dapat berisi "walk" atau "none" atau "others". Artinya jika "walk" berarti berjalan kaki, "none" jika rute tidak ditemukan dan "others" berarti menggunakan kendaran.

- Index ke 1 atau baris 8 pada listing 2.1 merupakan detail dari index 0. Artinya jika index 0 "walk" berarti index 1 harus "walk", "none" berarti index 1 harus "none" dan selain itu menyatakan id kendaraan yang mana bisa dipakai untuk ditampilkan gambarnya.

- Index ke 2 atau baris 9 pada listing 2.1 adalah array string yang berisi jalur dalam format "lat,lon". Maksud dari "lat,lon" disini adalah titik permulaan dan titik selesai.
- Index ke 3 atau bari 10 pada listing 2.1 merupakan berisi bentuk yang akan ditampilkan kepada pengguna. Informasi yang disampaikan dapat berupa:
  - %fromicon = untuk menunjukan ikon "from". Biasanya untuk mode presentasi di perangkat bergerak.
  - %toicon = untuk menunjukan ikon "to". Biasanya untuk mode presentasi di perangkat bergerak.
- Index ke 4 atau bari 11 pada listing 2.1 berisi URL untuk pemesanan tiket jika tersedia. Jika tidak tersedia akan bernilai null.

2.2.3 Web Service Pencarian Lokasi

Merupakan Kiri API yang digunakan untuk mencari lokasi beserta kordinat *latitude* dan *longitude*

Berikut parameter *request* yang diperlukan berikut penjelasanya:

version	2	Memberitahukan bahwa layanan yang dipakai adalah protokol veris 2
mode	"searchplace"	mengintruksikan layanan untuk mencari tempat
region	"cgk" or "bdo" or "sub"	kota yang akan dicari tempatnya
querystring	text apa saja dengan minimum text satu karakter	query string yang akan dicari menggunakan layanan ini
apikey	16-digit hexadecimals	API key yang digunakan

Berikut format Kiri API *responds*:

Listing 2.28: code *respond* pencarian lokasi

```
1  {
2      "status": "ok" or "error"
3      "searchresult": [
4          {
5              "placename": "place name"
6              "location": "lat,lon"
7          },
8          {
9              "placename": "place name"
10             "location": "lat,lon"
11         },
12         ...
13     ]
14     "attributions": [
15         "attribution_1", "attribution_2", ...
16     ]
17 }
```

Berikut maksud dari listing 2.2: Ketika pencarian lokasi sukses dilakukan maka status akan memberitahukan "ok" seperti di baris 2. Selanjutnya akan ditampilkan hasil dari lokasi yang ada beserta atributnya. Berikut keterangan dari format dari pencarian lokasi:

- Searchresult (pada bari 4 sampai 7, 8 sampai 11, dan seterusnya) berisi array dari tempat:
  - placename: nama tempat.

  - location: latitude dan longitude dari tempat.

- Attributions berisi array string yang berisikan atribut tambahan untuk dimunculkan.

2.2.4 Web Service Menemukan Transportasi Terdekat

Merupakan Kiri API yang digunakan untuk menemukan rute transportasi terdekat sesuai titik yang diinginkan pengguna.

Berikut parameter *request* yang diperlukan berikut penjelasanya:

version	2	Memberitahukan bahwa layanan yang dipakai adalah protokol veris 2
mode	"nearbytransports"	mengintruksikan layanan untuk mencari rute transportasi terdekat
start	latitude dan longitude (keduanya menggunakan nilai desimal)	kota yang akan dicari tempatnya
apikey	16-digit hexadecimals	API key yang digunakan

Berikut format Kiri API *responds*:

Listing 2.29: code *respond* menemukan lokasi terdekat

```
1  {
2      "status": "ok" or "error"
3      "nearbytransports": [
4          [
5              "walk" or "none" or others,
6              "walk" or vehicle_id or "none",
7              text string,
8              decimal value
9          ],
10         [
11             "walk" or "none" or others,
12             "walk" or vehicle_id or "none",
13             text string,
14             decimal value
15         ],
16         ...
17     ]
18 }
```

Berikut maksud dari listing 2.3: Ketika pencarian rute sukses dilakukan maka status akan memberitahukan "ok" seperti di baris 2. Selanjutnya akan diberikan array yang berisi transportasi terdekat yang diurutkan dari yang terdekat ke yang terjauh. Berikut keterangan dari setiap array tersebut:

- Index ke 0 atau baris 5 pada listing 2.1 dapat berisi "walk" atau "none" atau "others". Artinya jika "walk" berarti berjalan kaki, "none" jika rute tidak ditemukan dan "others" berarti menggunakan kendaran.

- Index ke 1 atau baris 6 pada listing 2.1 merupakan detail dari index 0. Artinya jika index 0 "walk" berarti index 1 harus "walk", "none" berarti index 1 harus "none" dan selain itu menyatakan id kendaraan yang mana bisa dipakai untuk ditampilkan gambarnya.

- Index ke 2 atau baris 7 pada listing 2.1 berisi nama kendaraan.

- Index ke 3 atau bari 8 pada listing 2.1 berisi jarak dengan satuan kilometer.

DAFTAR REFERENSI

[1] Manning, Paul *Pro Windows Phone App Development* 2013: Apress.

[2] Szostak, Tomasz *Windows Phone 8 Application Development Essentials* 2013: PACKT.