

**GUIDEBEE**  
**Digital Map API Developer Guide**  
**On Java ME platform**



James Shen  
[www.guidebee.biz](http://www.guidebee.biz)  
Guidebee Biz.

**ISSUE/AMENDMENT STATUS**

Issue	Date	Description	Author
1.0	24 <sup>th</sup> Jan 2009	First Version	James Shen <a href="mailto:jing.shen@guidebee.biz">jing.shen@guidebee.biz</a>

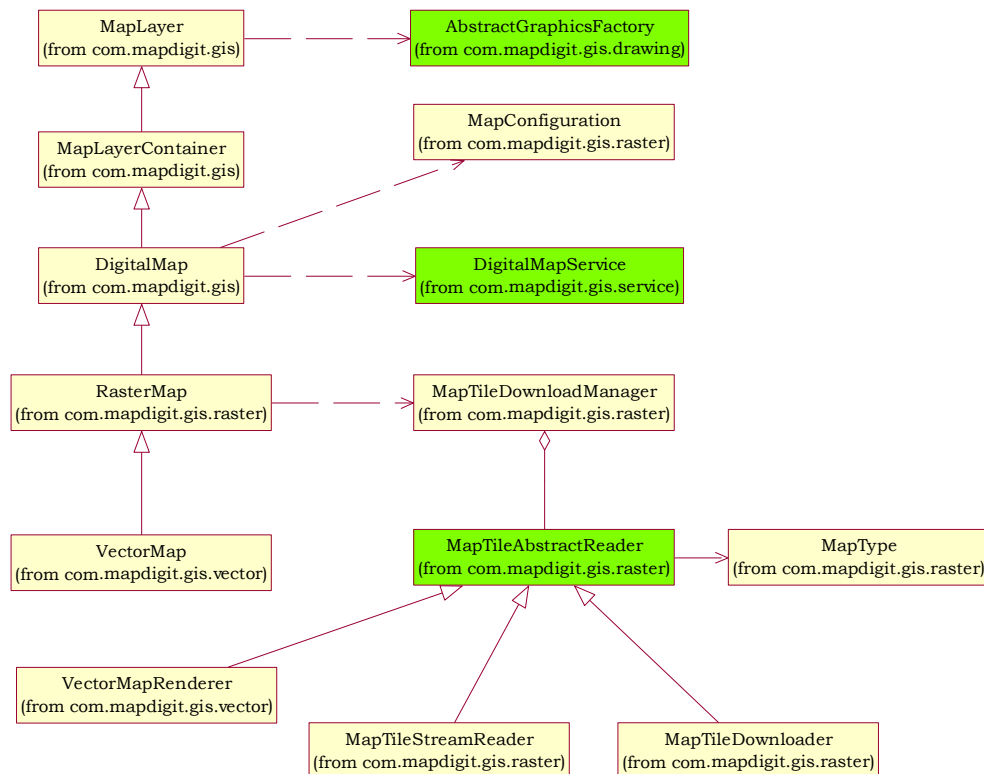
## TABLE OF CONTENTS

<b>ISSUE/AMENDMENT STATUS .....</b>	<b>2</b>
<b>1.0 OVERVIEW OF GUIDELEE DIGITAL MAP JAVA ME API .....</b>	<b>4</b>
1.1 PACKAGE GIS .....	5
1.2 PACKAGE GIS.GEOMETRY .....	6
1.3 PACKAGE GIS.DRAWING .....	7
1.4 PACKAGE GIS.RASTER.....	8
1.5 PACKAGE GIS.SERVICE.....	9
1.6 PACKAGE GIS.VECTOR .....	10
1.7 PACKAGE DRAWING.....	11
1.8 PACKAGE DRAWING.GEOMETRY .....	12
1.9 TYPICAL SOFTWARE ARCHITECTURE .....	13
<b>2.0 MAP BASIC.....</b>	<b>14</b>
2.1 MAP CLASS HIERARCHY.....	14
2.2 MAP TILE READER HIERARCHY .....	14
2.3 MAP SERVICE PROVIDER.....	15
2.4 GRAPHICS SUBSYSTEM .....	16
<b>3.0 RASTER MAP .....</b>	<b>17</b>
3.1 THE "HELLO, WORLD" OF RASTERMAP .....	17
3.1.1 Define a canvas to draw the map.....	18
3.1.2 Setup the Graphics System.....	18
3.1.3 Define a map downloader object to download map tiles from Server .....	19
3.1.4 Define the RasterMap Object.....	19
3.1.5 Initializing the Map.....	19
3.1.6 Define a map downloader callback to monitor the downloading progress .....	19
<b>4.0 STORED MAP .....</b>	<b>20</b>
<b>5.0 VECTOR MAP .....</b>	<b>23</b>
5.1 SAMPLE MAP DATA.....	23
5.2 RASTER MAPS .....	26
5.3 VECTOR MAPS.....	26
5.4 SAMPLE CODE.....	27
<b>6.0 MAP OPERATION .....</b>	<b>31</b>
6.1 SET MAP TYPE .....	31
6.2 ZOOM IN/ZOOM OUT .....	33
6.3 MAP PAN.....	35
6.4 MAP IMAGE CACHE .....	37
<b>7.0 MAP SERVICES .....</b>	<b>38</b>
7.1 GEOCODING .....	38
7.2 DRIVING DIRECTION.....	40
<b>8.0 PUT ALL THINGS TOGETHER –MOBILE LIVE MAP.....</b>	<b>43</b>
<b>9.0 LICENCE .....</b>	<b>45</b>

## 1.0 Overview of Guidebee Digital Map Java ME API

Guidebee Digital Map Java ME API implements a mobile GIS engine on J2ME platform (CLDC/MIDP). It support online, offline, raster and vector map in the same package.

Below is the core classes defined in the Map API.



Major features:

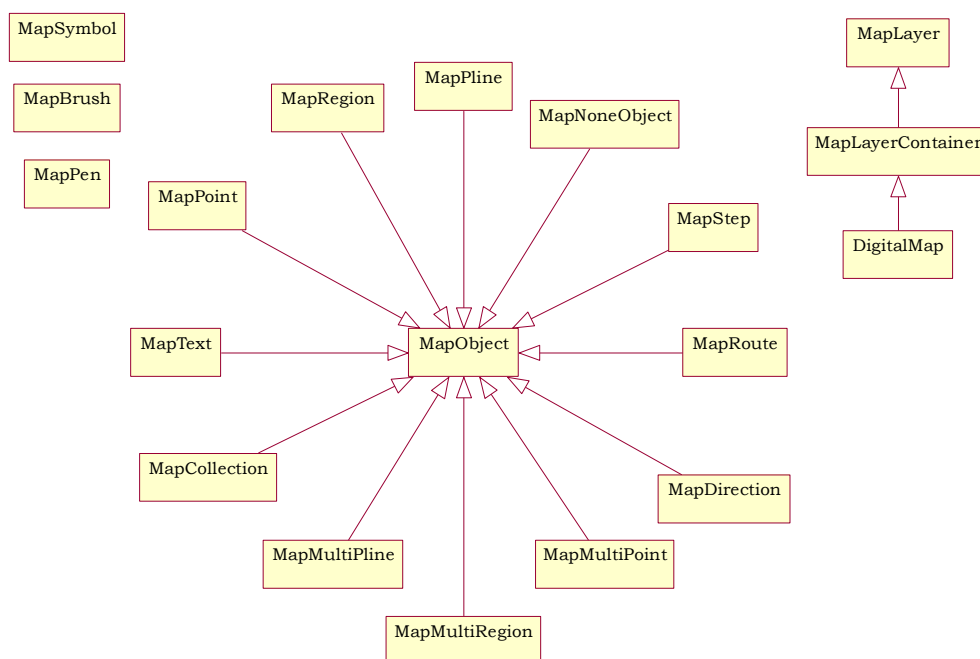
- Online map, it connects to map servers (like Yahoo map, Microsoft Live Map etc) and display map image.
- Offline map (or stored map), allow access self defined map image file locally to avoid data traffic to server.
- Vector map, it supports MapInfo compatible map data and provide similar functionalities as MapInfo map product.
- Multiple map layer support, it can easily add, remove a map layer to the Digital map.
- Consistent interface for online map, stored map and vector map. Panning and zooming.
- Map Service support (Geocoding, Reverse Geocoding, and routing).
- Great extensibility, the classes in green are extension points, they can be easily extended to support different Graphics System, Map service and more map types.
- 2D Graphics support, a high-performance 2D graphics API is included.

Digital Map APIs is provided in the following packages

- `com.mapdigit.collections`
- `com.mapdigit.drawing`
- `com.mapdigit.drawing.geometry`
- `com.mapdigit.gis`
- `com.mapdigit.gis.drawing`
- `com.mapdigit.gis.raster`
- `com.mapdigit.gis.service`
- `com.mapdigit.gis.vector`
- `com.mapdigit.util`
- `com.mapdigit.licence`

## 1.1 Package gis

Package gis define some common Map Objects:



MapDirection	This class is used to store driving directions results
MapObject	Base class of all map objects.
MapPoint	Class MapPoint stands for a point map object.
MapRoute	Objects of this class store information about a single route in a directions result.

MapStep	Objects of this class store information about a single step within a route in a directions result.
DigitalMap	DigitalMap is the base class for Raster Map and Vector Map..
MapSymbol	Map symbol used to display a point.
MapPen	Map pen used to draw a map object.
MapBrush	Map brush used to paint a map object.
MapPoint	Class MapPoint stands for a point map object.
MapPline	Class MapPline stands for a map pline object.
MapRegion	Class MapRegion stands for a map region object.
MapMultiPoint	Class MapMultiPoint stands for map points' collection.
MapMultiPline	Class MapMultiPline stands for map plines' collection.
MapMultiRegion	Class MapMultiRegion stands for map regions' collection.
MapCollection	Class MapCollection stands for a collection of map objects.
MapNoneObject	Class MapNoneObject stands for a map object without geo info.
MapText	Class MapText stands for a text map object.
MapLayer	Base class for a map layer.
MapLayerContainer	A container for map layers, support add, move, delete map layers.

## 1.2 Package gis.geometry

This package defines geographical geometry objects, like polygon, point, polyline etc.

GeoPoint

GeoBounds

GeoLatLng

GeoLatLngBounds

GeoPolyline

GeoSize

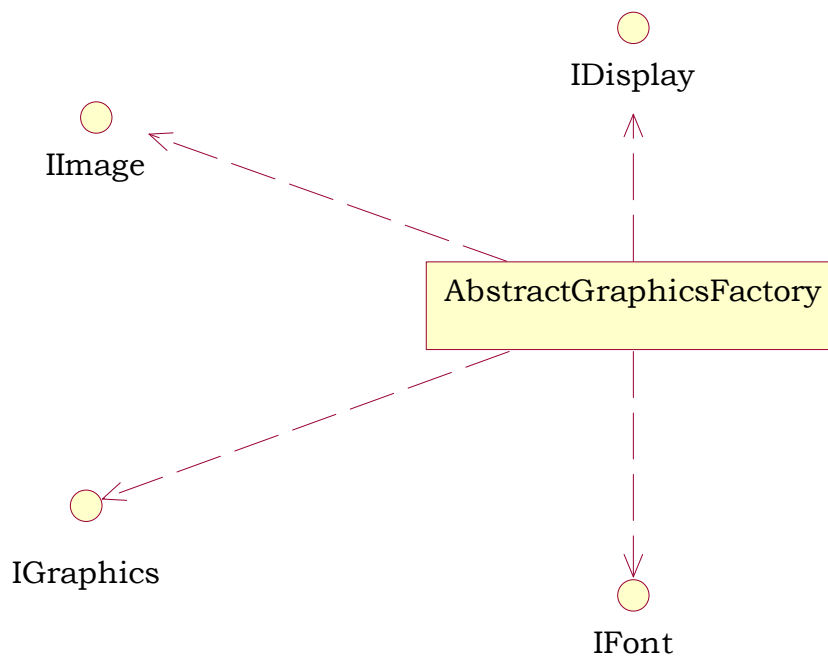
GeoPolygon

GeoBounds	GeoBounds is a rectangular area of the map in pixel coordinates
GeoLatLng	GeoLatLng is a point in geographical coordinate's longitude and latitude.
GeoLatLngBounds	GeoLatLng is a bound in geographical coordinate's longitude and latitude.
GeoPoint	A point representing a location in (x, y) coordinates space, specified in integer precision.
GeoPolygon	Polygon on map.

GeoPolyline	Polyline on map.
GeoSize	The GeoSize class encapsulates the width and height of a component (in integer precision) in a single object.

### 1.3 Package gis.drawing

This package define a common interface for different Graphics System, such as MIDP Graphics or LWUIT Graphics classes, with these interfaces ,Guidebee Digital Map can be used with different graphics systems.

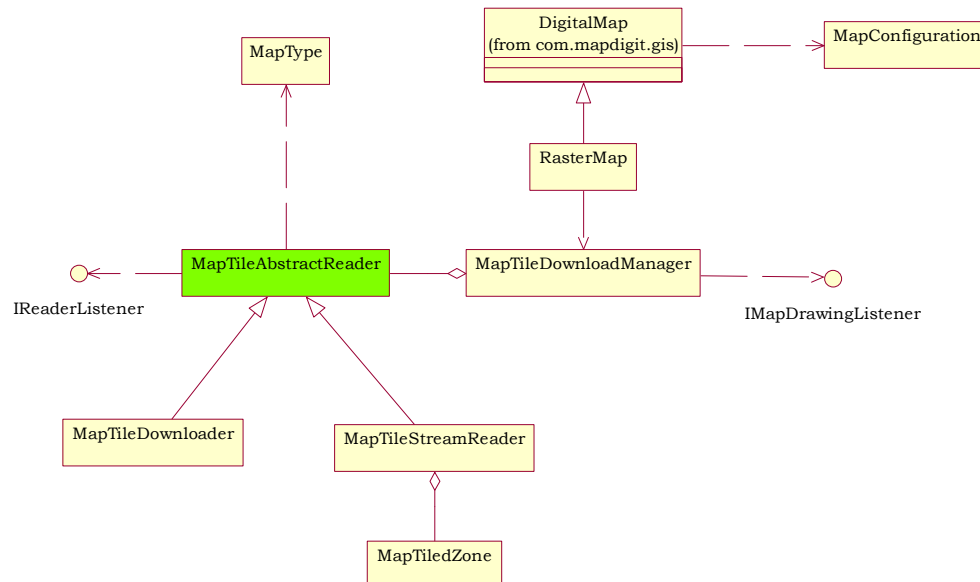


AbstractGraphcsFactory	Factor class, used to create other graphics objects like font, graphics ,images etc.
IDisplay	Physical display object interface.
IGraphics	Interface for Graphics object used to draw font, images etc, .
IFont	Font interface.
IImage	Image interface.

**Note:** these interfaces only define a small subset of corresponding Graphics, Font, Display, and Image classes defined in MIDP or LWUIT. These interface methods are used in GIS map render. With these interface, it's very easy to switch from one graphics system (say MIDP) to another graphics system (say LWUIT or any other graphics system on J2ME).

## 1.4 Package gis.raster

The raster package defines object used to raster maps (online or stored map).

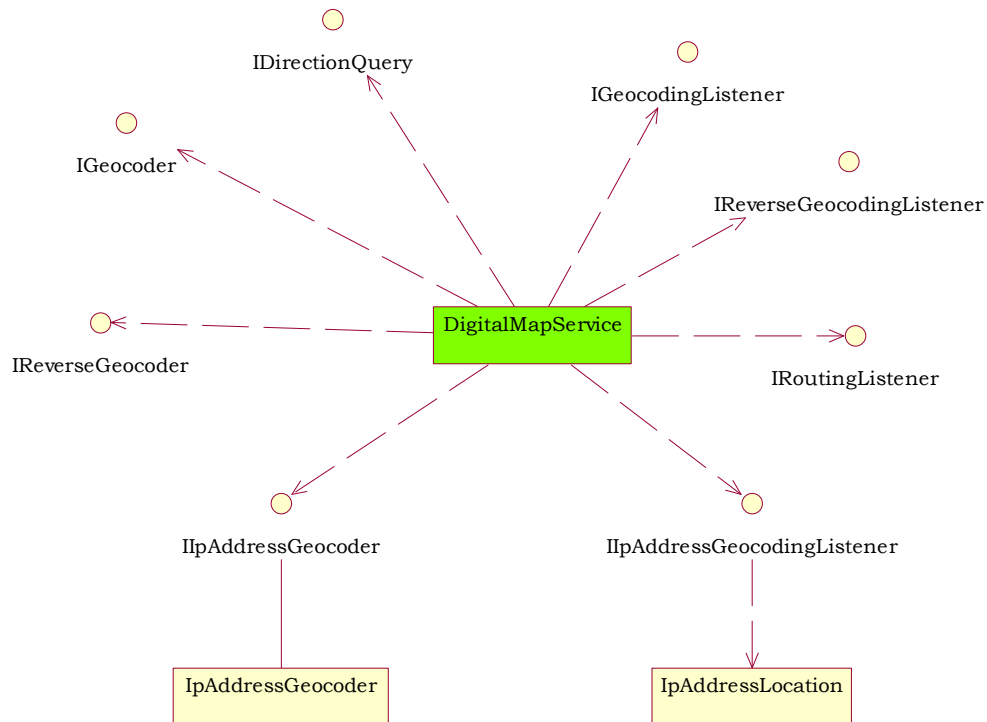


Map Type	Define different map types (Google, Microsoft Live, Yahoo etc).
MapConfiguration	To fit for different device capability (memory size, screen size), the Map API can be configured to turn on/off cache, drawing routing polyline or not etc).these configuration will effect memory usage and performance of the Map API.
RasterMap	A core map class used to display map from server or from stored map file, provide panning, zooming etc.
MapTileAbstractReader	An abstract class defined how map tiles is obtained, connecting to map server, reading from local map files or rendering the vector map.
MapTileDownloader	MapTileDownloader download map image tiles from server (msn, yahoo, etc).
MapTileStreamReader	Read map image tiles from a input stream (mostly from local map file)
MapTileZone	A predefined stored map file. Developer can define their own stored map file format, if only it derives from MapTileAbstractReader.
IReaderListener	A listener used to monitor the progress of the reading
IMapDrawingListener	When a map reader finish downloading/reading/rendering a map tile, it trigs the listener to notify a map tile is available.



## 1.5 Package gis.service

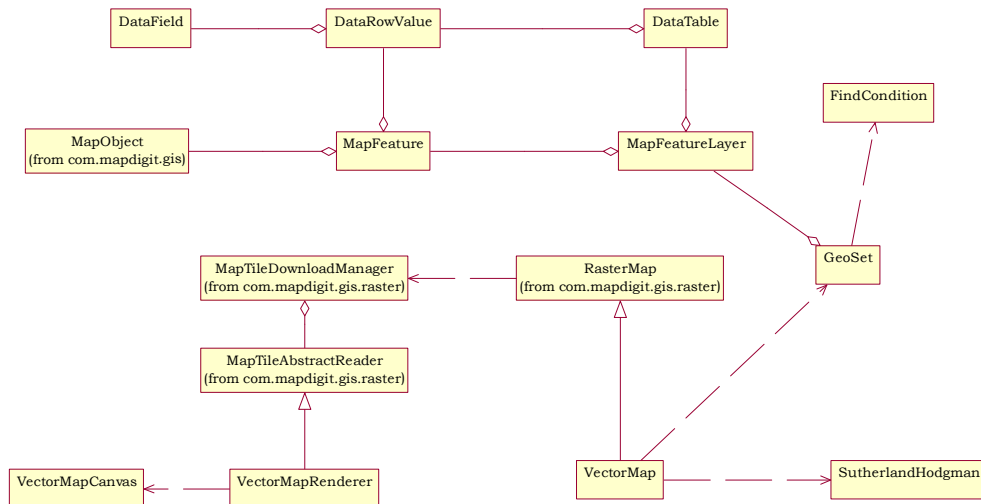
The service defines interface for Digital map services (Geocoding, reverse geocoding and routing etc).



DigitalMapServer	Default implementation for Map Service (routing, geocoding ,reverse geocoding)
IDirectionQuery	Interface to query routing information..
IGeocoder	Interface to find address.
IReverseGeocoder	Interface to find address based on its latitude and longitude.
IIpAddressGeocoder	Interface to find address based on its ip address (may not included in this release).
IGeocodingListener	Call back when geocoding is done.
IReverseGeocodingListener	Callback when reverse geocoding is done.
IRoutingListener	Callback when routing is done.
IIpAddressGeocodingListener	Callback when IP geocoding is done.
IpAddressGeocoder	Default implementation of IP geocoding service.

## 1.6 Package gis.vector

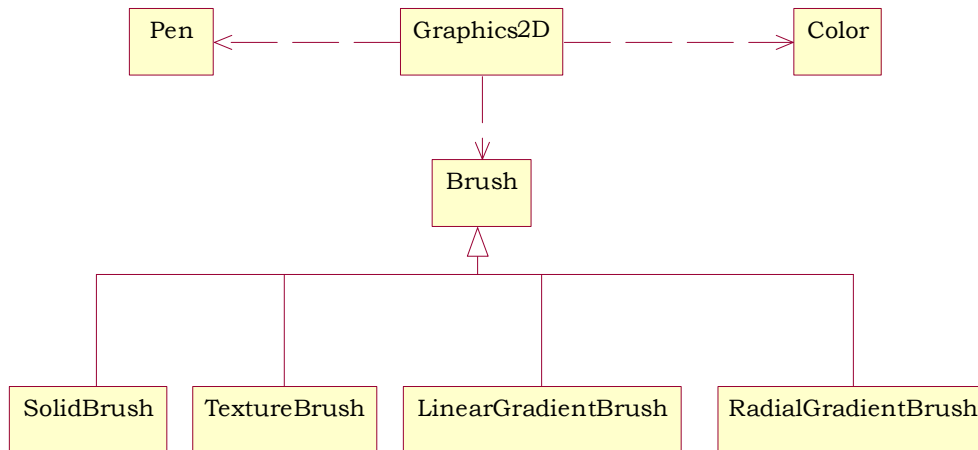
Vector package define objects for vector map.



SutherlandHodgman	Implements Sutherland-Hodgman clip algorithm.
VectorMap	VectorMap is the basic building blocks for Guidebee vector map. Each map is consists of multiple map Layers.
GeoSet	The Geoset object is built off the Map object and allows you to define a geoset. A Geoset is a collection of map layers and their settings.
MapFeatureLayer	MapLayer defines a map layer.Computer maps are organized into layers.
MapFeature	MapFeature defines a map feature in a map layer.
DataTable	Defines one tabular database table
DataField	Defines a field of a database table.
DataRowValue	Defines a row of a database table.
FindCondition	Defines a find condition when search for records.
FindConditions	Defines a find condition collection when search for records.

## 1.7 Package drawing

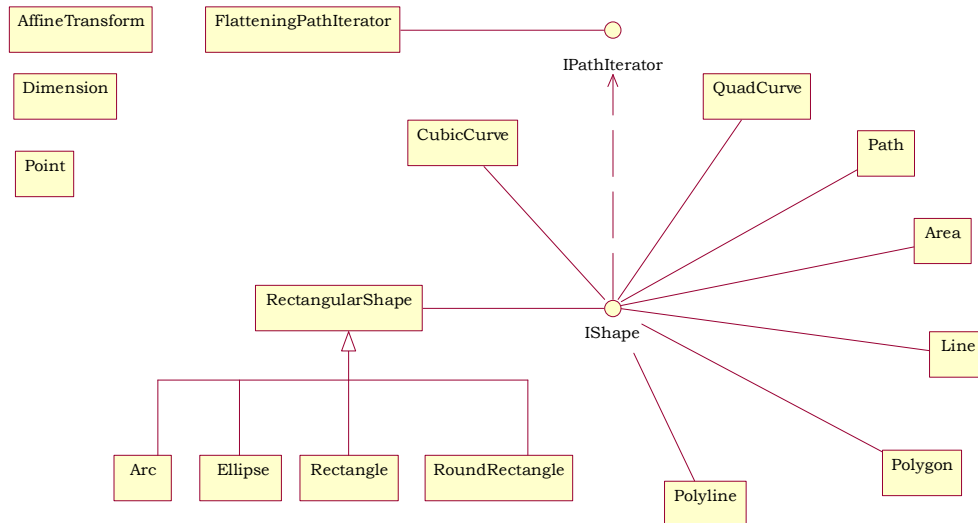
The Drawing package provides access to basic 2D graphics functionality



Brush	Classes derived from this abstract base class define objects used to fill the interiors of graphical shapes such as rectangles, ellipses, pies, polygons, and paths.
Color	The Color class is used to encapsulate colors in the default serge color space. Every color has an implicit alpha value of 1.0 or an explicit one provided in the constructor.
Graphics2D	This Graphics2D class provides more sophisticated control over geometry, coordinate transformations, color management, and text layout.
LinearGradientBrush	The LinearGradientBrush class provides a way to fill a Shape with a linear color gradient pattern.
Pen	The Pen class defines a basic set of rendering attributes for the outlines of graphics primitives, which are rendered with a Graphics2D object that has its Stroke attribute set to this Pen.
RadialGradientBrush	The RadialGradientBrush class provides a way to fill a shape with a circular radial color gradient pattern.
SolidBrush	Defines a brush of a single color.
TextureBrush	The TextureBrush class provides a way to fill a Shape with a texture that is specified as an Image.

## 1.8 Package drawing.geometry

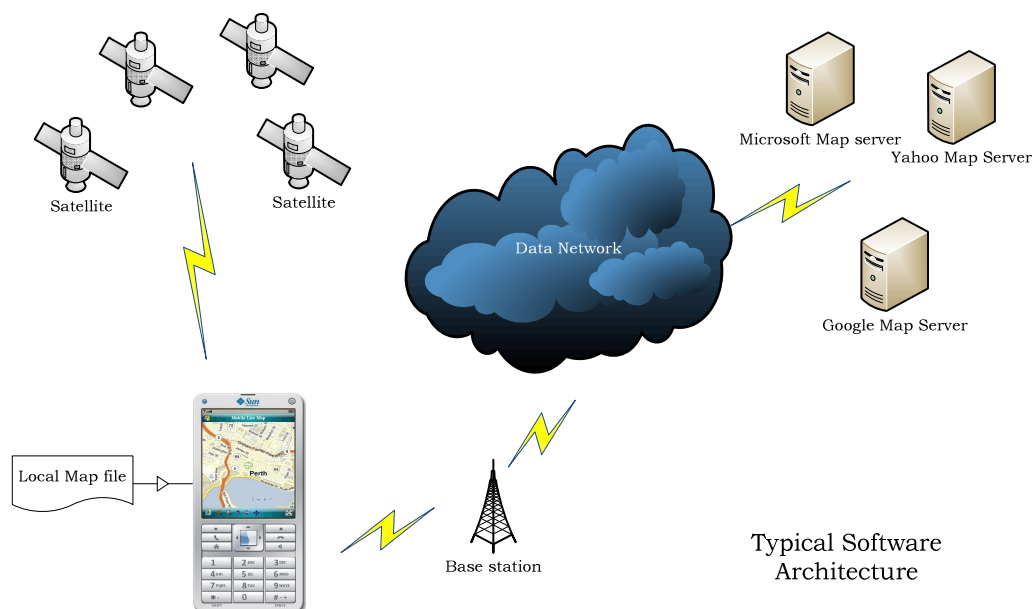
Package geometry provides the Java 2D classes for defining and performing operations on objects related to two-dimensional geometry.



<b>PathIterator</b>	The PathIterator interface provides the mechanism for objects that implement the Shape interface to return the geometry of their boundary by allowing a caller to retrieve the path of that boundary a segment at a time.
<b>IShape</b>	The Shape interface provides definitions for objects that represent some form of geometric shape.
<b>AffineTransform</b>	The AffineTransform class represents a 2D affine transform that performs a linear mapping from 2D coordinates to other 2D coordinates that preserves the "straightness" and "parallelness" of lines.
<b>Arc</b>	Arc store a 2D arc defined by a framing rectangle, start angle, angular extent (length of the arc), and a closure type (OPEN, CHORD, or PIE).
<b>Area</b>	An Area object stores and manipulates a resolution-independent description of an enclosed area of 2-dimensional space.
<b>CubicCurve</b>	The CubicCurve class defines a cubic parametric curve segment in (x, y) coordinate space.
<b>Dimension</b>	The Dimension class encapsulates the width and height of a component (in integer precision) in a single object.
<b>Ellipse</b>	The Ellipse class describes an ellipse that is defined by a framing rectangle.
<b>FlatteningPathIterator</b>	The FlatteningPathIterator class returns a flattened view of another PathIterator object.
<b>Path</b>	The Path class represents a geometric path constructed from straight

	lines, and quadratic and cubic (Bezier) curves.
<b>Line</b>	This Line represents a line segment in (x, y) coordinate space.
<b>Point</b>	A point representing a location in (x, y) coordinates space, specified in integer precision.
<b>Polygon</b>	The Polygon class encapsulates a description of a closed, two-dimensional region within a coordinate space.
<b>Polyline</b>	The Polyline class encapsulates a description of a collection of line segments within a coordinate space.
<b>QuadCurve</b>	The QuadCurve class defines a quadratic parametric curve segment in (x, y) coordinate space.
<b>Rectangle</b>	A Rectangle specifies an area in a coordinate space that is enclosed by the Rectangle object's upper-left point (x, y) in the coordinate space, its width, and its height.
<b>RectangularShape</b>	RectangularShape is the base class for a number of Shape objects whose geometry is defined by a rectangular frame.
<b>RoundRectangle</b>	The RoundRectangle class defines a rectangle with rounded corners defined by a location (x, y), a dimension (w x h), and the width and height of an arc with which to round the corners.

## 1.9 Typical Software architecture

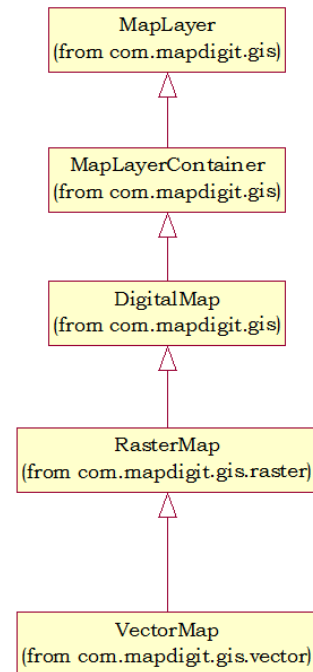


## 2.0 Map Basic

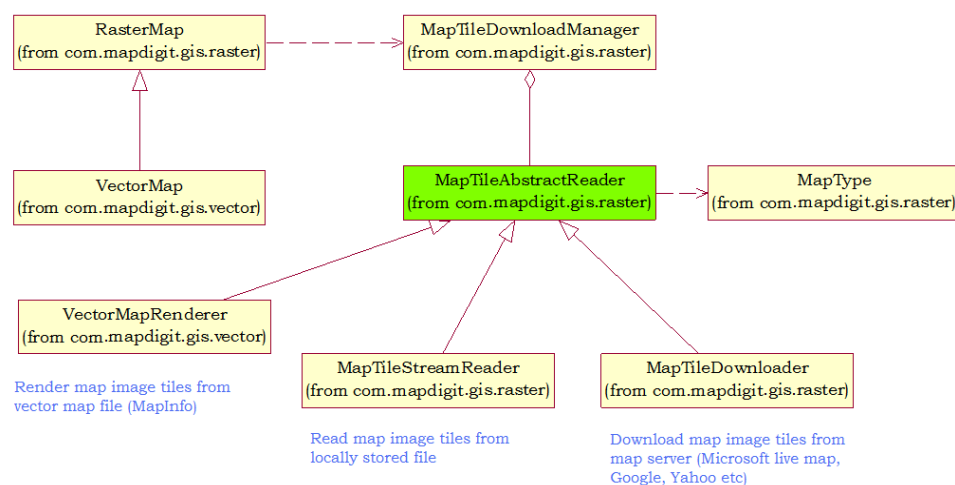
Guidebee Map API provide consistent interface for online, stored and vector map.

### 2.1 Map class hierarchy

- MapLayer define common map operation, like setCenter, Zoom in, Zoom out, Pan Direction and coordinates conversion (from Map to to screen and vice versa) and it's a common base class for all digital map.
- MapLayerContainer is container classes; manage a collection of Map layers allow them setCenter, Zoom In, Zoom out and Pan Direction together.
- DigitalMap introduces Map Services like Geocoding, Reverse Geocoding and Routing service. It supports incorporate different Map Services implementation.
- RasterMap deals with online, stored map images tiles, it support multi-threads and also provide listener to monitor the progress of downloading or reading map image tiles.
- VectorMap is a subclass of RasterMap, so it supports all functions provided by RasterMap, instead of downloading image tiles from map server or reading map image tiles from local files, Vector renders map image tile from MapInfo compatible vector map file also supports Geoset which managers multiple map feature layers.

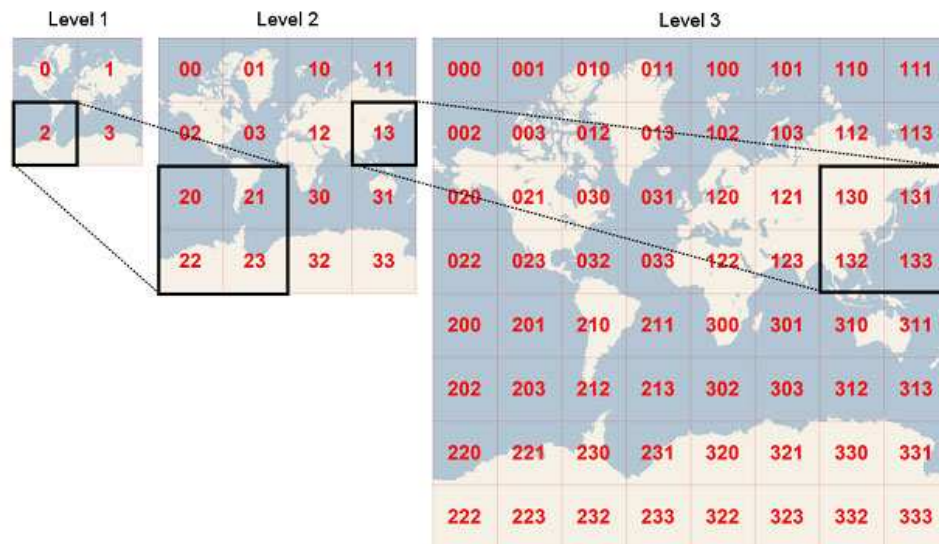


### 2.2 Map tile reader hierarchy



RasterMap has a reference to MapTileDownloadManager, which manages all map tile reader work threads.

Most map servers like Google, Microsoft, Yahoo uses follow map tiles system, these map server divides the whole map into 256X256 map tiles.



- MapTileAbstractReader is the common base class for all map tile readers, from the Rastermap prospective, there's no difference for online map, stored map or vector map, they have each MapTileAbstractReader subclass to handle the differences for online map, stored map and vector map.
- VectorMapRender renders map image tiles from MapInfo compatible vector map data file.
- MapTileStreamReader reads map image tiles from an input stream (mostly from a local stored file).
- MapTileDownloader downloads map image tiles from map servers like Google Map, Microsoft Live Map, and Yahoo Map etc.
- MapDirectionRenderer renders routing direction map tile, Guidebee Map API internally use and is not exposed as public API.

**Note:** For developer want to support self-defined stored map or connects to different map server, the only things is to subclass MapTileAbstractReader and implements related interface and then Guidebee Map API will works smoothly with your self-defined stored map or your own map server.

### 2.3 Map service provider

DigitalMap class has a reference to DigitalMapService, Digital Map Service provide geocoding, reverse geocoding and routing service. It's easy to switch map service providers for DigitalMap, for example, switch from Google map service to Microsoft map service or from local routing/geocoding service.

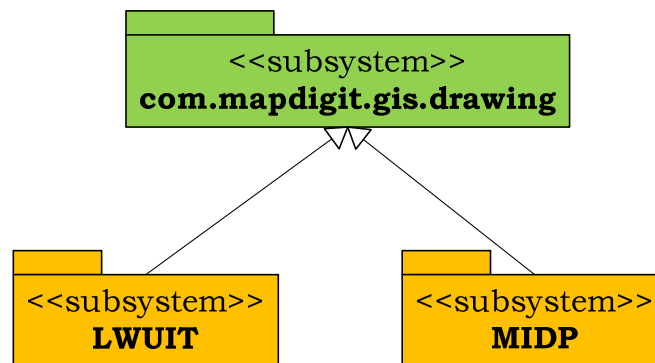


The current default map services are google map service (subject to change).



## 2.4 Graphics Subsystem

To make Guidebee Map API fit for different graphics sub systems on Java ME platform, an abstract Graphics Interface are defined in `com.mapdigit.gis.drawing` package.



The demo code includes two sample concrete implementation for the `com.mapdigit.gis.drawing` package. One for standard MIDP library, one for Sun's LWUIT library.



LWUIT Hello world demo  
Microsoft Live Map



MIDP Hello world demo  
Google Map

[Sun's LWUIT](#) is one of the best UI toolkits available on Java ME platform and it's free, so it's highly recommended to work with Guidebee Map API. But for low-tier or mid-tier phone where java heap size is below 2 mega byte, Standard MIDP is recommended to be used with Guidebee Map API.

Note: The minimum Java heap size for Guidebee Map API is 1 mega bytes. Phone which has less than 1M java heap size will encounter `OutOfMemoryError` when start Guidebee Map API applications.



### 3.0 Raster Map

Before using Guidebee Map API, a concrete implementation for all interfaces defined in `com.mapdigit.gis.drawing` is required, the `GISEngineTutorial` includes two default implementations, one for standard MIDP `lcdui` and one for Sun's `LWUIT`.

### 3.1 The "Hello, World" of RasterMap

The following example displays a map center at James Street, Perth, Australia.

```
package com.pstreets.gisengine.demo.midp;

import com.mapdigit.gis.MapLayer;
import com.mapdigit.gis.drawing.IGraphics;
import com.mapdigit.gis.drawing.IImage;
import com.mapdigit.gis.geometry.GeoLatLng;
import com.mapdigit.gis.raster.IMapDrawingListener;
import com.mapdigit.gis.raster.IReaderListener;
import com.mapdigit.gis.raster.MapTileDownloadManager;
import com.mapdigit.gis.raster.MapType;
import com.mapdigit.gis.raster.RasterMap;
import com.pstreets.gisengine.demo.MapDemoMIDlet;
import com.pstreets.gisengine.demo.drawing.midp.MIDPGraphicsFactory;
import javax.microedition.lcdui.Canvas;
import javax.microedition.lcdui.Display;
import javax.microedition.lcdui.Graphics;
import javax.microedition.lcdui.Image;

class MapCanvas extends Canvas{
    RasterMap map=null;

    MapCanvas(RasterMap map){
        this.map=map;
    }

    private void drawMapImage(Graphics g){
        IImage
        mapImage=MapLayer.getAbstractGraphicsFactory().createImage(this.getWidth(),
        this.getHeight());
        IGraphics mapGraphics=mapImage.getGraphics();
        map.paint(mapGraphics);
        g.drawImage((Image)mapImage.getNativeImage(), 0, 0, 0);
    }
    protected void paint(Graphics g) {
        drawMapImage(g);
    }
}

public class HelloWorldMIDP extends MapDemoMIDlet implements
IReaderListener, IMapDrawingListener{

    private MapCanvas canvas;
    private RasterMap map;

    private MapTileDownloadManager mapTileDownloadManager;

    public void startApp() {

        //set the graphics factory
        MIDPGraphicsFactory.getInstance().midlet=this;
    }
}
```

```

MapLayer.setAbstractGraphicsFactory(MIDPGraphicsFactory.getInstance());

    //Create the Digital Map objects.
    mapTileDownloadManager=new MapTileDownloadManager(this);
    mapTileDownloadManager.start();
    map=new RasterMap(512,512,mapTileDownloadManager);
    map.setMapDrawingListener(this);
    GeoLatLng center=new GeoLatLng(-31.948275,115.857562);
    map.setCenter(center,13, MapType.GOOGLEMAP);
    canvas=new MapCanvas(map);
    Display.getDisplay(this).setCurrent(canvas);
}

public void pauseApp() {
}

public void destroyApp(boolean unconditional) {
}

public void readProgress(int arg0, int arg1) {
    System.out.println(arg0+"/"+arg1);
}

public void done() {
    if(canvas!=null)
        canvas.repaint();
}
}

```

Even in this simple example, there are six things to note:

- Define a canvas to draw the map.
- Setup the Graphics System.
- Define a map downloader instance to download the map tiles from Map Server.
- Define a RasterMap instance to create a new “Map” object.
- Center the map on a given geographic point.
- Define a map downloader callback to monitor the downloading progress.

These steps are explained below.

### 3.1.1 Define a canvas to draw the map

In the example, we define a MapCanva which extends from Canvas. It takes ServerMap as parameter for the constructor.

In it's paint methods

```

IGraphics mapGraphics=mapImage.getGraphics();
map.paint(mapGraphics);
g.drawImage((Image)mapImage.getNativeImage(), 0, 0, 0);

```

### 3.1.2 Setup the Graphics System

The sample code uses MIDP's lcdui as it's graphics system. The following code

```
MIDPGraphicsFactory.getInstance().midlet=this;
```

```
MapLayer.setAbstractGraphicsFactory(MIDPGraphicsFactory.getInstance());
```

Setup MIDP lcdui as GIS rendering graphics system.

If want to use LWUIT, use following code

```
//set the graphics factory
MapLayer.setAbstractGraphicsFactory(LWUITGraphicsFactory.getInstance());
```

### 3.1.3 Define a map downloader object to download map tiles from Server

RasterMap needs a map downloader object to download map tiles from map server. The Guidebee Map API provides a default MapDownloader object which can be used to download map tiles from Yahoo,Microsoft etc map server. You can define your own map downloader to get map tiles from other servers.

To monitor the downloader progress, a MapdownloaderListerner is needed for the MapDownloader.

### 3.1.4 Define the RasterMap Object.

ServerMap's constructor is defined as following:

```
public ServerMap(int width,
                 int height,
                 MapTileDownloader downloader)
    throws biz.guidebee.licence.InvalidLicenceException
```

width and height defines the width and height of the map ,normally it's the width and height of the canvas.

### 3.1.5 Initializing the Map

```
GeoLatLng center=new GeoLatLng(-31.948275,115.857562);
map.setCenter(center,13,mapTileDownloader.MICROSOFTMAP);
```

setCenter() method requires a GLatLng coordinate and a zoom level and this method must be sent before any other operations are performed on the map.

setCenter() also specify the map type, it can be Yahoo ,Microsoft Map or Satellite Map etc. In this example we use Microsoft Live Map.

### 3.1.6 Define a map downloader callback to monitor the downloading progress

Downloading map tiles from Map server takes times (it depends on the network connection speed) ,it'd be good let user know the downloading is going on. MapDownloaderListener defines one callback readProgress gives the total bytes to be downloaded and the bytes already downloaded, in this example, we just print it out.

```
public void readProgress(int bytes,int total){
    System.out.println("Reading " + bytes + "/" + total);
}
```

Note: the super class MapDemoMIDlet is a direct subclass of MIDlet; it just feeds the correct licence information for using Guidee Map API.

## 4.0 Stored Map

There's no major difference for stored map and online map, they are all raster map. The only difference is Stored Map uses MapTileStreamReader reads map tiles from locally stored file while online map uses MapTileDownloader downloads map tiles from map server. Developer can write his/her own MapTileStreamReader to read self defined Map Tile file to work with Guidebee Map API.

```
package com.pstreets.gisengine.demo.midp;

import com.mapdigit.gis.MapLayer;
import com.mapdigit.gis.drawing.IGraphics;
import com.mapdigit.gis.drawing.IImage;
import com.mapdigit.gis.geometry.GeoLatLng;
import com.mapdigit.gis.raster.IMapDrawingListener;
import com.mapdigit.gis.raster.IReaderListener;
import com.mapdigit.gis.raster.MapTileDownloadManager;
import com.mapdigit.gis.raster.MapTileStreamReader;
import com.mapdigit.gis.raster.MapTiledZone;
import com.mapdigit.gis.raster.MapType;
import com.mapdigit.gis.raster.RasterMap;
import com.pstreets.gisengine.demo.MapDemoMIDlet;
import com.pstreets.gisengine.demo.drawing.midp.MIDPGraphicsFactory;
import java.io.ByteArrayInputStream;
import java.io.DataInputStream;
import java.io.IOException;
import java.io.InputStream;
import javax.microedition.lcdui.Canvas;
import javax.microedition.lcdui.Display;
import javax.microedition.lcdui.Graphics;
import javax.microedition.lcdui.Image;

class StoredMapCanvas extends Canvas{
    RasterMap map=null;

    StoredMapCanvas(RasterMap map){
        this.map=map;
    }

    private void drawMapImage(Graphics g){
        IImage
mapImage=MapLayer.getAbstractGraphicsFactory().createImage(this.getWidth(),
this.getHeight());
        IGraphics mapGraphics=mapImage.getGraphics();
        map.paint(mapGraphics);
        g.drawImage((Image)mapImage.getNativeImage(), 0, 0, 0);
    }
    protected void paint(Graphics g) {
        drawMapImage(g);
    }
}

public class StoredMapMIDP extends MapDemoMIDlet implements
IReaderListener, IMapDrawingListener{

    private StoredMapCanvas canvas;
    private RasterMap map;
```

```

private MapTileDownloadManager mapTileDownloadManager;

public void startApp() {

    //set the graphics factory
    MIDPGraphicsFactory.getInstance().midlet=this;

    MapLayer.setAbstractGraphicsFactory(MIDPGraphicsFactory.getInstance());

    InputStream is =
this.getClass().getResourceAsStream("/world03.map");
    byte[] buffer = null;
    try {
        buffer = new byte[is.available()];
        is.read(buffer);
        is.close();
    } catch (IOException ex) {
        ex.printStackTrace();
    }
    ByteArrayInputStream bais = new ByteArrayInputStream(buffer);
    MapTiledZone mapTiledZone = new MapTiledZone(new
DataInputStream(bais));

    MapTileStreamReader localMapTileFileReader = new
MapTileStreamReader();
    localMapTileFileReader.addZone(mapTiledZone);
    try {
        localMapTileFileReader.open();
    } catch (Exception ex) {
        ex.printStackTrace();
    }

    mapTileDownloadManager = new MapTileDownloadManager(this,
localMapTileFileReader);
    mapTileDownloadManager.start();
    map=new RasterMap(512,512,mapTileDownloadManager);
    map.setMapDrawingListener(this);
    GeoLatLng center=new GeoLatLng(-31.948275,115.857562);
    map.setCenter(center,3, MapType.GOOGLEMAP);
    canvas=new StoredMapCanvas(map);
    Display.getDisplay(this).setCurrent(canvas);
}

public void pauseApp() {
}

public void destroyApp(boolean unconditional) {
}

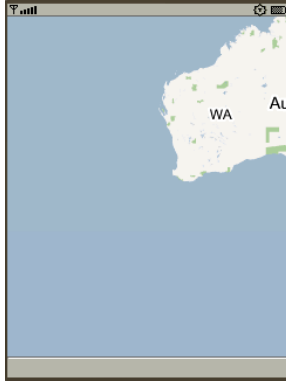
public void readProgress(int arg0, int arg1) {
    System.out.println(arg0+"/"+arg1);
}

public void done() {
    if(canvas!=null)
        canvas.repaint();
}
}

```

The difference is the code in blue, stored map read map tiles from local stored map, here is world03.map. And MapTileDownloadManager's constructor uses MapTileAbstractReader as its second input parameter.

If you want to define your own map tile file format, subclass from MapTileAbstractReader and implements public abstract void getImage(int mtype, int x, int y, int zoomLevel); which read map tile data based on the index for map tiles (x,y and zoomLevel).

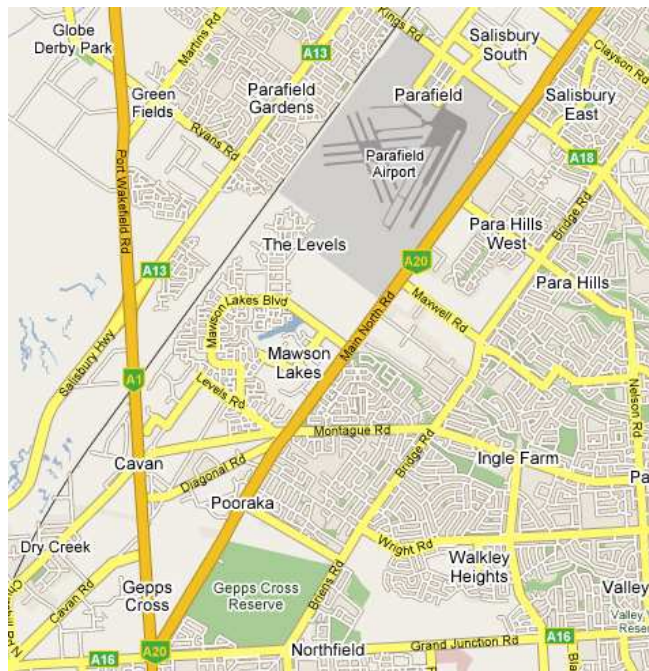


## 5.0 Vector Map

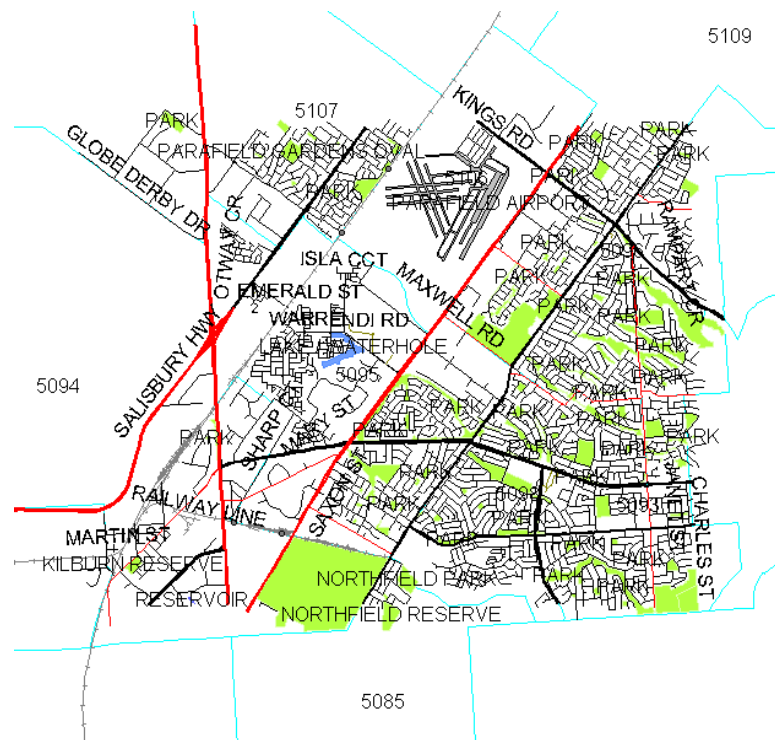
Vector map in some way can be considered as a raster map, because in the end ,the vector map need to be rendered as images on screen). So similar as stored map, there's no major difference use vector map with online/offline map. The default vector map implementation uses MapInfo compatible data format.

### 5.1 Sample Map Data

In this developer guide, we'll use map data of Mawson Lakes, South Australia, following is the google map clip of such area.



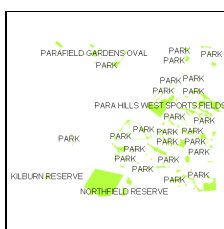
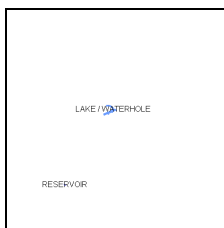

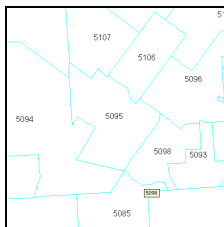
The MapInfo display such area as following



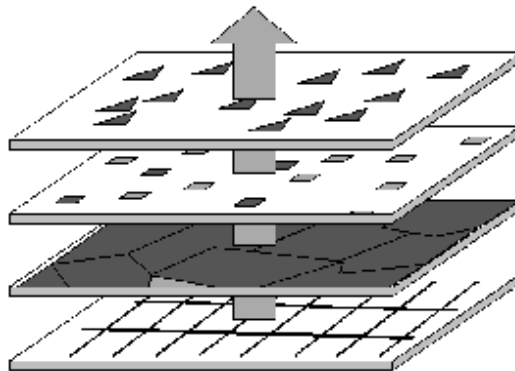
The original map data contains 7 map layers:

<



Park	<table><tr><th>NAME</th><th>TYPE</th><th>STE</th><th>SP_UFI</th><th>MIFCODE</th></tr><tr><td>PARAFIELD GARDENS OVAL</td><td>803</td><td>4</td><td>93,072</td><td>60,010,110</td></tr><tr><td>PARK</td><td>803</td><td>4</td><td>94,457</td><td>60,010,110</td></tr><tr><td>PARK</td><td>803</td><td>4</td><td>94,458</td><td>60,010,110</td></tr><tr><td>PARK</td><td>803</td><td>4</td><td>94,459</td><td>60,010,110</td></tr><tr><td>PARK</td><td>803</td><td>4</td><td>94,513</td><td>60,010,110</td></tr></table>	NAME	TYPE	STE	SP_UFI	MIFCODE	PARAFIELD GARDENS OVAL	803	4	93,072	60,010,110	PARK	803	4	94,457	60,010,110	PARK	803	4	94,458	60,010,110	PARK	803	4	94,459	60,010,110	PARK	803	4	94,513	60,010,110	
NAME	TYPE	STE	SP_UFI	MIFCODE																												
PARAFIELD GARDENS OVAL	803	4	93,072	60,010,110																												
PARK	803	4	94,457	60,010,110																												
PARK	803	4	94,458	60,010,110																												
PARK	803	4	94,459	60,010,110																												
PARK	803	4	94,513	60,010,110																												
Water body	<table><tr><th>NAME</th><th>TYPE</th><th>STE</th><th>SP_UFI</th><th>MIFCODE</th></tr><tr><td>RESERVOIR</td><td>105</td><td>4</td><td>2,190,194</td><td>30,030,300</td></tr><tr><td>LAKE / WATERHOLE</td><td>104</td><td>4</td><td>2,193,306</td><td>30,030,100</td></tr></table>	NAME	TYPE	STE	SP_UFI	MIFCODE	RESERVOIR	105	4	2,190,194	30,030,300	LAKE / WATERHOLE	104	4	2,193,306	30,030,100																
NAME	TYPE	STE	SP_UFI	MIFCODE																												
RESERVOIR	105	4	2,190,194	30,030,300																												
LAKE / WATERHOLE	104	4	2,193,306	30,030,100																												
Airport	<table><tr><th>NAME</th><th>TYPE</th><th>STE</th><th>SP_UFI</th><th>MIFCODE</th></tr><tr><td>PARAFIELD AIRPORT</td><td>700</td><td>4</td><td>75,115</td><td>10,310,101</td></tr></table>	NAME	TYPE	STE	SP_UFI	MIFCODE	PARAFIELD AIRPORT	700	4	75,115	10,310,101																					
NAME	TYPE	STE	SP_UFI	MIFCODE																												
PARAFIELD AIRPORT	700	4	75,115	10,310,101																												
Post code region	<table><tr><th>Postcode</th><th>Locality</th></tr><tr><td>5109</td><td>Multiple Localities - See Index</td></tr><tr><td>5106</td><td>Multiple Localities - See Index</td></tr><tr><td>5085</td><td>Multiple Localities - See Index</td></tr><tr><td>5095</td><td>Multiple Localities - See Index</td></tr><tr><td>5096</td><td>Multiple Localities - See Index</td></tr><tr><td>5098</td><td>Multiple Localities - See Index</td></tr><tr><td>5094</td><td>Multiple Localities - See Index</td></tr></table>	Postcode	Locality	5109	Multiple Localities - See Index	5106	Multiple Localities - See Index	5085	Multiple Localities - See Index	5095	Multiple Localities - See Index	5096	Multiple Localities - See Index	5098	Multiple Localities - See Index	5094	Multiple Localities - See Index															
Postcode	Locality																															
5109	Multiple Localities - See Index																															
5106	Multiple Localities - See Index																															
5085	Multiple Localities - See Index																															
5095	Multiple Localities - See Index																															
5096	Multiple Localities - See Index																															
5098	Multiple Localities - See Index																															
5094	Multiple Localities - See Index																															

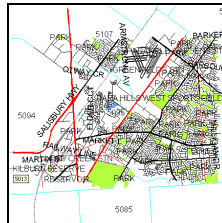
We have already said that computer maps are a collection of layers. Each map table that contains graphic objects can be displayed as a layer in a map window. For example, you can a table of streets, and a table of county boundaries.



Think of these layers as transparencies where each layer contains a different part of the map. The layers are stacked one on top of the other and allow you to see all aspects of the map at the same time. For example, one layer may contain country boundaries, a second

layer may have symbols that represent capitals, and a third layer might consist of highways. Laying these transparencies one on top of the other builds a complete map.

So laying above 7 map layers of Mawson Lakes, South Australia. We get the final map contains railway station, streets, airport, parks etc.



Unlike google map, which is of raster map type, Guidebee local map API provides vector map support as MapInfo does.

There are two basic types of digital maps, and it's important to understand the difference between the two. The first is Raster maps, and the second is Vector maps.

## 5.2 Raster Maps



Raster maps are based on conventional image files. These maps are, quite simply, scans of a paper maps that are saved in an image format like .gif or .png. They are generated by running the original topo maps through a large scanner and saving the copy as a digital file. The advantage of this type of map? They are an exact duplicate of the original paper topo map, with all of the detail that we're used to seeing.

The disadvantage? Being an image file (just like a digital photo), they don't scale well. If you try to "zoom in" on the image, it quickly becomes pixellated and fuzzy. If you "zoom out" to any significant degree, the text and details on the map quickly becomes unreadable.

Because raster maps retain all of the detail of the map upon which they are based, they are a fairly large image file.

## 5.3 Vector Maps



Vector maps are not an image file like raster maps. Vector maps consist of a text file with coordinates describing the various points and curves on a map. When they are loaded into a mapping program, that data is used to generate a map image.

As you would expect, the advantages and disadvantages of this map type are the exact opposite of those of raster maps. Because the image is generated "on the fly" from the data, these maps scale up and down with no loss of clarity. You can zoom out to see an entire river system, or zoom in to see specific areas and retain the readability of the map in both.

The only disadvantage to vector maps might be the diminished level of detail. Although there is no reason that the vector data could contain all of the detail of an original topo, these maps almost always have less detail.

## 5.4 Sample code

```

package com.pstreets.gisengine.demo.midp;

import com.mapdigit.gis.MapLayer;
import com.mapdigit.gis.drawing.IFont;
import com.mapdigit.gis.drawing.IGraphics;
import com.mapdigit.gis.drawing.IImage;
import com.mapdigit.gis.geometry.GeoLatLng;
import com.mapdigit.gis.raster.IMapDrawingListener;
import com.mapdigit.gis.raster.IReaderListener;
import com.mapdigit.gis.raster.MapTileDownloadManager;
import com.mapdigit.gis.raster.MapType;
import com.mapdigit.gis.raster.RasterMap;
import com.mapdigit.gis.vector.GeoSet;
import com.mapdigit.gis.vector.MapFeatureLayer;
import com.mapdigit.gis.vector.VectorMap;
import com.mapdigit.gis.vector.VectorMapRenderer;
import com.pstreets.gisengine.demo.MapDemoMIDlet;
import com.pstreets.gisengine.demo.drawing.midp.MIDPGraphicsFactory;
import java.io.ByteArrayInputStream;
import java.io.DataInputStream;
import java.io.IOException;
import java.io.InputStream;
import javax.microedition.lcdui.Canvas;
import javax.microedition.lcdui.Display;
import javax.microedition.lcdui.Font;
import javax.microedition.lcdui.Graphics;
import javax.microedition.lcdui.Image;

class VectorMapCanvas extends Canvas{
    RasterMap map=null;

    VectorMapCanvas(RasterMap map){
        this.map=map;
    }

    private void drawMapImage(Graphics g){
        IImage
mapImage=MapLayer.getAbstractGraphicsFactory().createImage(this.getWidth(),
this.getHeight());
        IGraphics mapGraphics=mapImage.getGraphics();
        map.paint(mapGraphics);
        g.drawImage((Image)mapImage.getNativeImage(), 0, 0, 0);
    }
    protected void paint(Graphics g) {
        drawMapImage(g);
    }
}
/**
 * @author james
 */

public class VectorMapMIDP extends MapDemoMIDlet implements
IReaderListener,IMapDrawingListener{

    private VectorMapCanvas canvas;
    private RasterMap map;

```

```

private MapTileDownloadManager mapTileDownloadManager;

public void startApp() {

    //set the graphics factory
    MIDPGraphicsFactory.getInstance().midlet=this;

    MapLayer.setAbstractGraphicsFactory(MIDPGraphicsFactory.getInstance());
    GeoSet geoSet=null;
    //Create the Digital Map objects.
    try {
        InputStream geoStream =
this.getClass().getResourceAsStream("/map/MawsonLakes.pst");
        InputStream airStream =
this.getClass().getResourceAsStream("/map/AUS_AIR.lyr");
        InputStream drStream =
this.getClass().getResourceAsStream("/map/AUS_DR.lyr");
        InputStream pkStream =
this.getClass().getResourceAsStream("/map/AUS_PK.lyr");
        InputStream rlStream =
this.getClass().getResourceAsStream("/map/AUS_RL.lyr");
        InputStream rlsStream =
this.getClass().getResourceAsStream("/map/AUS_RLS.lyr");
        InputStream stStream =
this.getClass().getResourceAsStream("/map/AUS_ST.lyr");
        InputStream pcStream =
this.getClass().getResourceAsStream("/map/AUST05_PC.lyr");

        byte[] bufferGeo = null;
        bufferGeo = new byte[geoStream.available()];
        geoStream.read(bufferGeo);
        geoStream.close();
        ByteArrayInputStream baisGeo = new
ByteArrayInputStream(bufferGeo);
        geoSet = new GeoSet(new DataInputStream(baisGeo));

        byte[] bufferAIR = null;
        bufferAIR = new byte[airStream.available()];
        airStream.read(bufferAIR);
        airStream.close();
        ByteArrayInputStream baisAIR = new
ByteArrayInputStream(bufferAIR);
        MapFeatureLayer layerAIR=new MapFeatureLayer(new
DataInputStream(baisAIR));
        //layerAIR.fontColor=0xFF0000;

        byte[] bufferDR = null;
        bufferDR = new byte[drStream.available()];
        drStream.read(bufferDR);
        drStream.close();
        ByteArrayInputStream baisDR = new
ByteArrayInputStream(bufferDR);
        MapFeatureLayer layerDR=new MapFeatureLayer(new
DataInputStream(baisDR));
        //layerDR.fontColor=0x00FF00;

        byte[] bufferPK = null;
        bufferPK = new byte[pkStream.available()];
        pkStream.read(bufferPK);
        pkStream.close();
        ByteArrayInputStream baisPK = new
ByteArrayInputStream(bufferPK);
    }
}

```

```

        MapFeatureLayer layerPK=new MapFeatureLayer(new
DataInputStream(baisPK));
        //.fontColor=0x0000FF;

        byte[] bufferRL = null;
        bufferRL = new byte[rlStream.available()];
        rlStream.read(bufferRL);
        rlStream.close();
        ByteArrayInputStream baisRL = new
ByteArrayInputStream(bufferRL);
        MapFeatureLayer layerRL=new MapFeatureLayer(new
DataInputStream(baisRL));
        //layerRL.fontColor=0xFFFF00;

        byte[] bufferRLS = null;
        bufferRLS = new byte[rlsStream.available()];
        rlsStream.read(bufferRLS);
        rlsStream.close();
        ByteArrayInputStream baisRLS = new
ByteArrayInputStream(bufferRLS);
        MapFeatureLayer layerRLS=new MapFeatureLayer(new
DataInputStream(baisRLS));
        //layerRLS.fontColor=0xFF00FF;

        byte[] bufferST = null;
        bufferST = new byte[stStream.available()];
        stStream.read(bufferST);
        stStream.close();
        ByteArrayInputStream baisST = new
ByteArrayInputStream(bufferST);
        MapFeatureLayer layerST=new MapFeatureLayer(new
DataInputStream(baisST));
        //layerST.fontColor=0x00FFFF;

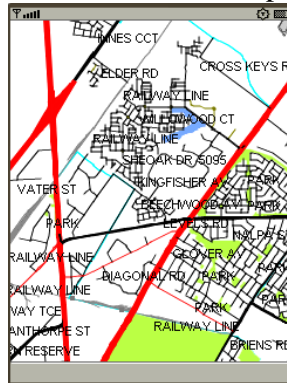
        byte[] bufferPC = null;
        bufferPC = new byte[pcStream.available()];
        pcStream.read(bufferPC);
        pcStream.close();
        ByteArrayInputStream baisPC = new
ByteArrayInputStream(bufferPC);
        MapFeatureLayer layerPC=new MapFeatureLayer(new
DataInputStream(baisPC));
        //layerPC.fontColor=0x8080FF;

        geoSet.addMapFeatureLayer(layerRLS);
        geoSet.addMapFeatureLayer(layerST);
        geoSet.addMapFeatureLayer(layerRL);
        geoSet.addMapFeatureLayer(layerPK);
        geoSet.addMapFeatureLayer(layerDR);
        geoSet.addMapFeatureLayer(layerAIR);
        geoSet.addMapFeatureLayer(layerPC);

        VectorMapRenderrer vectorMapRenderrer=new
VectorMapRenderrer(geoSet);
        Font font=Font.getDefaultFont();
        IFont
newFont=MapLayer.getAbstractGraphicsFactory().createFont(font);
        vectorMapRenderrer.setFont(newFont);
        mapTileDownloadManager = new MapTileDownloadManager(this,
vectorMapRenderrer);
        geoSet.open();

```

The code in blue ,opens vector map and create a VectorMapRender to read the vector map, the rest of the code is the same as stored and online map.



## 6.0 Map Operation

ServerMap has methods to zoom/pan the map.

### 6.1 Set Map Type

In the “Hello, World” example, we specify the map type when calling `setCenter()`. We can also call `setMapType` to change the type of map be shown. From different map server (Yahoo, Microsoft etc) or (Normal Map, Satellite Map, Hybrid map).

The following example show different map type in sequence.

```
package com.pstreets.gisengine.demo.midp;

import com.mapdigit.gis.MapLayer;
import com.mapdigit.gis.drawing.IGraphics;
import com.mapdigit.gis.drawing.IImage;
import com.mapdigit.gis.geometry.GeoLatLng;
import com.mapdigit.gis.raster.IMapDrawingListener;
import com.mapdigit.gis.raster.IReaderListener;
import com.mapdigit.gis.raster.MapTileDownloadManager;
import com.mapdigit.gis.raster.MapType;
import com.mapdigit.gis.raster.RasterMap;
import com.pstreets.gisengine.demo.MapDemoMIDlet;
import com.pstreets.gisengine.demo.drawing.midp.MIDPGraphicsFactory;
import javax.microedition.lcdui.Canvas;
import javax.microedition.lcdui.Command;
import javax.microedition.lcdui.CommandListener;
import javax.microedition.lcdui.Display;
import javax.microedition.lcdui.Displayable;
import javax.microedition.lcdui.Graphics;
import javax.microedition.lcdui.Image;

class MapTypeCanvas extends Canvas implements CommandListener{
    RasterMap map=null;
    private int mapType=0;
    private static int []mapTypes={MapTileDownloader.MICROSOFTSATELLITE,
    MapTileDownloader.MICROSOFTMAP,
    MapTileDownloader.MICROSOFTHYBRID};

    private Command mapTypeCommand=new Command("MapType",Command.OK,1);

    MapTypeCanvas(RasterMap map){
        this.map=map;
        addCommand(mapTypeCommand);
        setCommandListener(this);
    }

    private void drawMapImage(Graphics g){
        IImage
        mapImage=MapLayer.getAbstractGraphicsFactory().createImage(this.getWidth(),
        this.getHeight());
        IGraphics mapGraphics=mapImage.getGraphics();
        map.paint(mapGraphics);
        g.drawImage((Image)mapImage.getNativeImage(), 0, 0, 0);
    }
    protected void paint(Graphics g) {
        drawMapImage(g);
    }
}
```

```

        public void commandAction(Command c, Displayable d) {
            if(c==mapTypeCommand){
                map.setMapType(mapTypes[mapType]);
                mapType++;
                mapType %= mapTypes.length;
            }
        }
    }
}
/**
 * @author james
 */

public class MapTypeMIDP extends MapDemoMIDlet implements
IReaderListener, IMapDrawingListener{

    private MapTypeCanvas canvas;
    private RasterMap map;

    private MapTileDownloadManager mapTileDownloadManager;

    public void startApp() {

        //set the graphics factory
        MIDPGraphicsFactory.getInstance().midlet=this;

        MapLayer.setAbstractGraphicsFactory(MIDPGraphicsFactory.getInstance());

        //Create the Digital Map objects.
        mapTileDownloadManager=new MapTileDownloadManager(this);
        mapTileDownloadManager.start();
        map=new RasterMap(512,512,mapTileDownloadManager);
        map.setMapDrawingListener(this);
        GeoLatLng center=new GeoLatLng(-31.948275,115.857562);
        map.setCenter(center,13, MapType.GOOGLEMAP);
        canvas=new MapTypeCanvas(map);
        Display.getDisplay(this).setCurrent(canvas);
    }

    public void pauseApp() {
    }

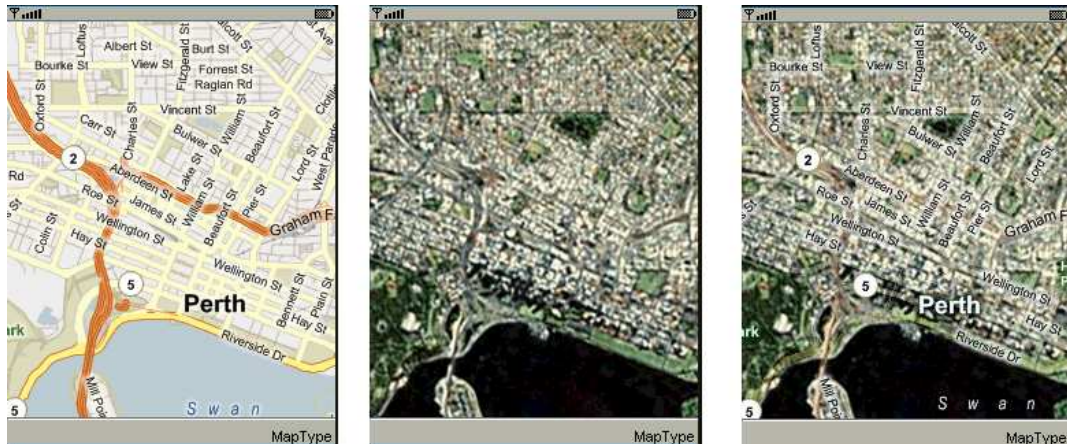
    public void destroyApp(boolean unconditional) {
    }

    public void readProgress(int arg0, int arg1) {
        System.out.println(arg0+"/"+arg1);
    }

    public void done() {
        if(canvas!=null)
            canvas.repaint();
    }
}

```





## 6.2 Zoom In/Zoom Out

The following examples display how to zoom in/Zoom out map.

```
package com.pstreets.gisengine.demo.midp;

import com.mapdigit.gis.MapLayer;
import com.mapdigit.gis.drawing.IGraphics;
import com.mapdigit.gis.drawing.IImage;
import com.mapdigit.gis.geometry.GeoLatLng;
import com.mapdigit.gis.raster.IMapDrawingListener;
import com.mapdigit.gis.raster.IReaderListener;
import com.mapdigit.gis.raster.MapTileDownloadManager;
import com.mapdigit.gis.raster.MapType;
import com.mapdigit.gis.raster.RasterMap;
import com.pstreets.gisengine.demo.MapDemoMIDlet;
import com.pstreets.gisengine.demo.drawing.midp.MIDPGraphicsFactory;
import javax.microedition.lcdui.Canvas;
import javax.microedition.lcdui.Command;
import javax.microedition.lcdui.CommandListener;
import javax.microedition.lcdui.Display;
import javax.microedition.lcdui.Displayable;
import javax.microedition.lcdui.Graphics;
import javax.microedition.lcdui.Image;

class MapZoomCanvas extends Canvas implements CommandListener{
    RasterMap map=null;

    private Command mapZoomInCommand=new Command("Zoom In",Command.OK,1);
    private Command mapZoomOutCommand=new Command("Zoom
    Out",Command.CANCEL,1);

    MapZoomCanvas(RasterMap map){
        this.map=map;
        addCommand(mapZoomInCommand);
        addCommand(mapZoomOutCommand);
        setCommandListener(this);
    }

    private void drawMapImage(Graphics g){
        IImage
        mapImage=MapLayer.getAbstractGraphicsFactory().createImage(this.getWidth(),
```

```

this.getHeight());
    IGraphics mapGraphics=mapImage.getGraphics();
    map.paint(mapGraphics);
    g.drawImage((Image)mapImage.getNativeImage(), 0, 0, 0);
}
protected void paint(Graphics g) {
    drawMapImage(g);
}

public void commandAction(Command c, Displayable d) {
    if(c==mapZoomInCommand){
        map.zoomIn();

    }else if(c==mapZoomOutCommand){
        map.zoomOut();
    }
}
}
/**
 * @author james
 */

public class MapZoomMIDP extends MapDemoMIDlet implements
IReaderListener, IMapDrawingListener{

    private MapZoomCanvas canvas;
    private RasterMap map;

    private MapTileDownloadManager mapTileDownloadManager;

    public void startApp() {

        //set the graphics factory
        MIDPGraphicsFactory.getInstance().midlet=this;

        MapLayer.setAbstractGraphicsFactory(MIDPGraphicsFactory.getInstance());

        //Create the Digital Map objects.
        mapTileDownloadManager=new MapTileDownloadManager(this);
        mapTileDownloadManager.start();
        map=new RasterMap(512,512,mapTileDownloadManager);
        map.setMapDrawingListener(this);
        GeoLatLng center=new GeoLatLng(-31.948275,115.857562);
        map.setCenter(center,13, MapType.GOOGLEMAP);
        canvas=new MapZoomCanvas(map);
        Display.getDisplay(this).setCurrent(canvas);
    }

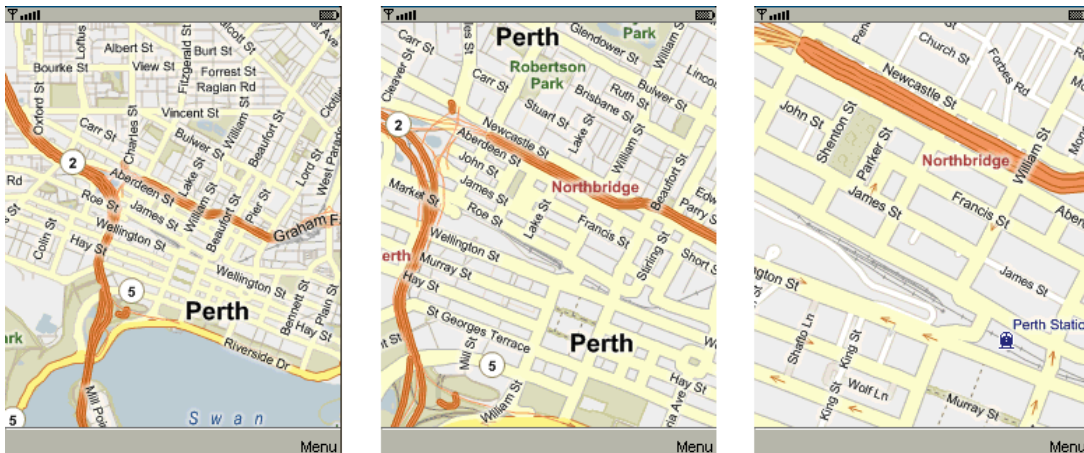
    public void pauseApp() {
    }

    public void destroyApp(boolean unconditional) {
    }

    public void readProgress(int arg0, int arg1) {
        System.out.println(arg0+"/"+arg1);
    }

    public void done() {
        if(canvas!=null)
            canvas.repaint();
    }
}

```



### 6.3 Map Pan

ServerMap has two methods, `panDirection` and `panTo`. `panTo` pan the current pan to a given latitude,longitude, `panDirection` move the map to some distance relative to current position.

The following example how to pan the map in up,down,left,right directions.

```
package com.pstreets.gisengine.demo.midp;

import com.mapdigit.gis.MapLayer;
import com.mapdigit.gis.drawing.IGraphics;
import com.mapdigit.gis.drawing.IImage;
import com.mapdigit.gis.geometry.GeoLatLng;
import com.mapdigit.gis.raster.IMapDrawingListener;
import com.mapdigit.gis.raster.IReaderListener;
import com.mapdigit.gis.raster.MapTileDownloadManager;
import com.mapdigit.gis.raster.MapType;
import com.mapdigit.gis.raster.RasterMap;
import com.pstreets.gisengine.demo.MapDemoMIDlet;
import com.pstreets.gisengine.demo.drawing.midp.MIDPGraphicsFactory;
import javax.microedition.lcdui.Canvas;
import javax.microedition.lcdui.Command;
import javax.microedition.lcdui.CommandListener;
import javax.microedition.lcdui.Display;
import javax.microedition.lcdui.Displayable;
import javax.microedition.lcdui.Graphics;
import javax.microedition.lcdui.Image;

class MapPanCanvas extends Canvas implements CommandListener{
    RasterMap map=null;

    private Command mapUpCommand=new Command("Up",Command.OK,1);
    private Command mapDownCommand=new Command("Down",Command.ITEM,1);
    private Command mapLeftCommand=new Command("Left",Command.ITEM,1);
    private Command mapRightCommand=new Command("Down",Command.ITEM,1);

    MapPanCanvas(RasterMap map){
```

```

        this.map=map;
        addCommand(mapUpCommand);
        addCommand(mapDownCommand);
        addCommand(mapLeftCommand);
        addCommand(mapRightCommand);
        setCommandListener(this);
    }

    private void drawMapImage(Graphics g){
        Image
mapImage=MapLayer.getAbstractGraphicsFactory().createImage(this.getWidth(),
this.getHeight());
        IGraphics mapGraphics=mapImage.getGraphics();
        map.paint(mapGraphics);
        g.drawImage((Image)mapImage.getNativeImage(), 0, 0, 0);
    }
    protected void paint(Graphics g) {
        drawMapImage(g);
    }

    public void commandAction(Command c, Displayable d) {
        if(c==mapUpCommand){
            map.panDirection(0, -32);

        }else if(c==mapDownCommand){
            map.panDirection(0, 32);
        }else if(c==mapLeftCommand){
            map.panDirection(-32, 0);
        }else if(c==mapRightCommand){
            map.panDirection(32, 0);
        }
        repaint();
    }
}
/**
 * @author james
 */

public class MapPanMIDP extends MapDemoMIDlet implements
IReaderListener,IMapDrawingListener{

    private MapPanCanvas canvas;
    private RasterMap map;

    private MapTileDownloadManager mapTileDownloadManager;

    public void startApp() {

        //set the graphics factory
        MIDPGraphicsFactory.getInstance().midlet=this;

        MapLayer.setAbstractGraphicsFactory(MIDPGraphicsFactory.getInstance());

        //Create the Digital Map objects.
        mapTileDownloadManager=new MapTileDownloadManager(this);
        mapTileDownloadManager.start();
        map=new RasterMap(512,512,mapTileDownloadManager);
        map.setMapDrawingListener(this);
        GeoLatLng center=new GeoLatLng(-31.948275,115.857562);
        map.setCenter(center,13, MapType.GOOGLEMAP);
        canvas=new MapPanCanvas(map);
        Display.getDisplay(this).setCurrent(canvas);
    }
}

```

```
public void pauseApp() {  
}  
  
public void destroyApp(boolean unconditional) {  
}  
  
public void readProgress(int arg0, int arg1) {  
    System.out.println(arg0+"/"+arg1);  
}  
  
public void done() {  
    if(canvas!=null)  
        canvas.repaint();  
}  
}
```

## 6.4 Map image cache

RasterMap has some internal cache to speed up future map tiles download. If the image required is already in the cache, it reads from the internal cache memory directly. But this cache is temporary; it's gone when application exits. If application wants to save the cache permanently, ServerMap provides two methods: `saveMapCache ()` and `restoreMapCache ()` to save and restore map image cache to and from a recordstore.

## 7.0 Map Services

Guidebee Map API also provides methods let application to access geocoding and routing services from Map Server.

### 7.1 GeoCoding

Geocoding is the process of converting addresses (like "1600 Amphitheatre Parkway, Mountain View, CA") into geographic coordinates (like latitude 37.423021 and longitude -122.083739), which you can use to place markers or position the map.

The following examples show how to use the Geocoding services with ServerMap, it queries for 7 Fairway, Crawley, Australia. And then display the map in that area.

```
package com.pstreets.gisengine.demo.midp;

import com.mapdigit.gis.MapLayer;
import com.mapdigit.gis.MapPoint;
import com.mapdigit.gis.drawing.IGraphics;
import com.mapdigit.gis.drawing.IImage;
import com.mapdigit.gis.geometry.GeoLatLng;
import com.mapdigit.gis.raster.IMapDrawingListener;
import com.mapdigit.gis.raster.IReaderListener;
import com.mapdigit.gis.raster.MapTileDownloadManager;
import com.mapdigit.gis.raster.MapType;
import com.mapdigit.gis.raster.RasterMap;
import com.mapdigit.gis.service.IGeocodingListener;
import com.pstreets.gisengine.demo.MapDemoMIDlet;
import com.pstreets.gisengine.demo.drawing.midp.MIDPGraphicsFactory;
import javax.microedition.lcdui.Canvas;
import javax.microedition.lcdui.Command;
import javax.microedition.lcdui.CommandListener;
import javax.microedition.lcdui.Display;
import javax.microedition.lcdui.Displayable;
import javax.microedition.lcdui.Graphics;
import javax.microedition.lcdui.Image;

class MapGeocodingCanvas extends Canvas implements
CommandListener, IGeocodingListener{
    RasterMap map=null;

    private Command mapFindAddressCommand=new Command("Find
Address",Command.OK,1);

    MapGeocodingCanvas(RasterMap map){
        this.map=map;
        addCommand(mapFindAddressCommand);
        map.setGeocodingListener(this);
        setCommandListener(this);
    }

    private void drawMapImage(Graphics g){
        IImage
mapImage=MapLayer.getAbstractGraphicsFactory().createImage(this.getWidth(),
this.getHeight());
        IGraphics mapGraphics=mapImage.getGraphics();
        map.paint(mapGraphics);
    }
}
```

```

        g.drawImage((Image)mapImage.getNativeImage(), 0, 0, 0);
    }
    protected void paint(Graphics g) {
        drawMapImage(g);
    }

    public void commandAction(Command c, Displayable d) {
        if(c==mapFindAddressCommand){
            map.getLocations("7 Fairway,Crawley,WA 6009,Australia");
        }
    }

    public void done(String arg0, MapPoint[] result) {
        if(result!=null){
            map.panTo(result[0].getPoint());
            repaint();
        }
    }

    public void readProgress(int arg0, int arg1) {
        System.out.println(arg0+"/"+arg1);
    }
}
/**
 * @author james
 */

public class MapGeocodingMIDP extends MapDemoMIDlet implements
IReaderListener,IMapDrawingListener{

    private MapGeocodingCanvas canvas;
    private RasterMap map;

    private MapTileDownloadManager mapTileDownloadManager;

    public void startApp() {

        //set the graphics factory
        MIDPGraphicsFactory.getInstance().midlet=this;

        MapLayer.setAbstractGraphicsFactory(MIDPGraphicsFactory.getInstance());

        //Create the Digital Map objects.
        mapTileDownloadManager=new MapTileDownloadManager(this);
        mapTileDownloadManager.start();
        map=new RasterMap(512,512,mapTileDownloadManager);
        map.setMapDrawingListener(this);
        GeoLatLng center=new GeoLatLng(-31.948275,115.857562);
        map.setCenter(center,13, MapType.GOOGLEMAP);
        canvas=new MapGeocodingCanvas(map);
        Display.getDisplay(this).setCurrent(canvas);
    }

    public void pauseApp() {
    }

    public void destroyApp(boolean unconditional) {
    }

    public void readProgress(int arg0, int arg1) {
        System.out.println(arg0+"/"+arg1);
    }
}

```



```

    public void done() {
        if(canvas!=null)
            canvas.repaint();
    }
}

```

Before calling getLocation(), a callback(or a listener) need to be setup to get the query result, ServerMap uses AJAX to make a web query and get the result with the done methods. The MapPoint[] array contains the geocoding results.

## 7.2 Driving Direction

You can get driving direction by the getDirection() method of ServerMap. Once directions are returned, the MapDirection object will internally store results which you can retrieve using MapDirection.getPolyline() and/or MapDirection.getRoute(i:Number) methods. Steps within a route can be retrieved using the MapRoute.getStep(i:Number)

Here is the example:

```

package com.pstreets.gisengine.demo.midp;

import com.mapdigit.gis.MapDirection;
import com.mapdigit.gis.MapLayer;
import com.mapdigit.gis.drawing.IGraphics;
import com.mapdigit.gis.drawing.IImage;
import com.mapdigit.gis.geometry.GeoLatLng;
import com.mapdigit.gis.raster.IMapDrawingListener;
import com.mapdigit.gis.raster.IReaderListener;
import com.mapdigit.gis.raster.MapTileDownloadManager;
import com.mapdigit.gis.raster.MapType;
import com.mapdigit.gis.raster.RasterMap;
import com.mapdigit.gis.service.IRoutingListener;
import com.pstreets.gisengine.demo.MapDemoMIDlet;
import com.pstreets.gisengine.demo.drawing.midp.MIDPGraphicsFactory;
import javax.microedition.lcdui.Canvas;
import javax.microedition.lcdui.Command;
import javax.microedition.lcdui.CommandListener;
import javax.microedition.lcdui.Display;
import javax.microedition.lcdui.Displayable;
import javax.microedition.lcdui.Graphics;
import javax.microedition.lcdui.Image;

class MapRoutingCanvas extends Canvas implements
CommandListener, IRoutingListener{
    RasterMap map=null;

    private Command mapGetDirectionCommand=new Command("Get
Direction",Command.OK,1);

    MapRoutingCanvas(RasterMap map){
        this.map=map;
        addCommand(mapGetDirectionCommand);
        map.setRoutingListener(this);
    }
}

```



```

        setCommandListener(this);
    }

    private void drawMapImage(Graphics g){
        Image
mapImage=MapLayer.getAbstractGraphicsFactory().createImage(this.getWidth(),
this.getHeight());
        IGraphics mapGraphics=mapImage.getGraphics();
        map.paint(mapGraphics);
        g.drawImage((Image)mapImage.getNativeImage(), 0, 0, 0);
    }
    protected void paint(Graphics g) {
        drawMapImage(g);
    }

    public void commandAction(Command c, Displayable d) {
        if(c==mapGetDirectionCommand){
            map.getDirections("from: 68 Waterloo St,Joondanna,WA
6060,Australia to: 7 Fairway,Crawley,WA 6009,Australia");
        }
    }

    public void readProgress(int arg0, int arg1) {
        System.out.println(arg0+"/"+arg1);
    }

    public void done(String arg0, MapDirection result) {
        if(result!=null){
            map.setMapDirection(result);
            map.resize(result.getBound());
        }
    }
}
/**
 * @author james
 */

public class MapRoutingMIDP extends MapDemoMIDlet implements
IReaderListener,IMapDrawingListener{

    private MapRoutingCanvas canvas;
    private RasterMap map;

    private MapTileDownloadManager mapTileDownloadManager;

    public void startApp() {

        //set the graphics factory
        MIDPGraphicsFactory.getInstance().midlet=this;

        MapLayer.setAbstractGraphicsFactory(MIDPGraphicsFactory.getInstance());

        //Create the Digital Map objects.
        mapTileDownloadManager=new MapTileDownloadManager(this);
        mapTileDownloadManager.start();
        map=new RasterMap(512,512,mapTileDownloadManager);
        map.setMapDrawingListener(this);
        GeoLatLng center=new GeoLatLng(-31.948275,115.857562);
        map.setCenter(center,13, MapType.GOOGLEMAP);
        canvas=new MapRoutingCanvas(map);
        Display.getDisplay(this).setCurrent(canvas);
    }
}

```

```
}

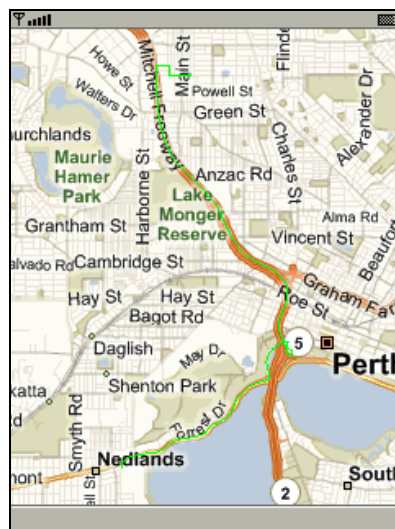
public void pauseApp() {
}

public void destroyApp(boolean unconditional) {
}

public void readProgress(int arg0, int arg1) {
    System.out.println(arg0+"/"+arg1);
}

public void done() {
    if(canvas!=null)
        canvas.repaint();
}
}
```

Note: Currently Guidebee Map API Geocoding and Direction Services make use of Google's web service. But it's subject to change in the future. But the API will keep the same.



## 8.0 Put All things together –Mobile Live Map

Microsoft's Pocket Streets is one of my favorite applications, though it doesn't provide navigation capability, but it's my first location software ever installed in my HP PDA. Now with Guidebee Map API, we can develop a similar application on Java ME platform.

It provides following features:

- Maps meant for mobile

Mobile Live Map is designed for sharp, colorful imaging on your device display. Zoom and pan to easily navigate your route.

- Points of interest

Quickly find or add your own points of interest, such as ATMs, restaurants, or transportation.

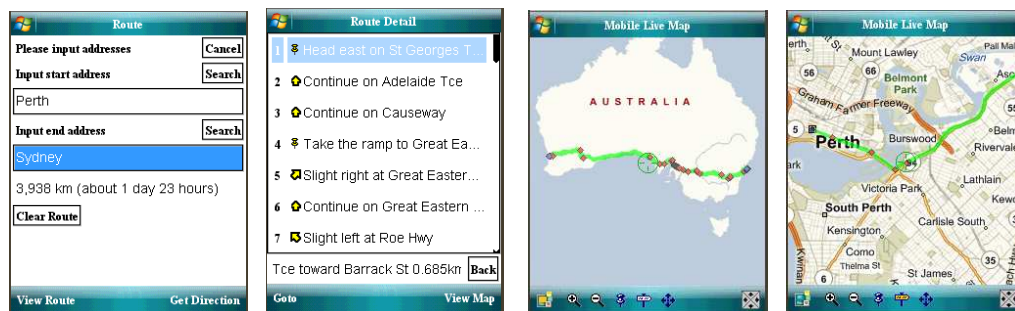
- Find addresses and places

Search for destinations and addresses using partial names and "roam" maps to find points of interest.

- GPS support

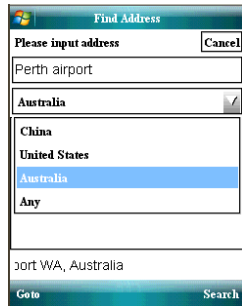
Use Mobile Live Map plus your compatible GPS device to indicate your real-time location on a map.

### Route



### Map Type



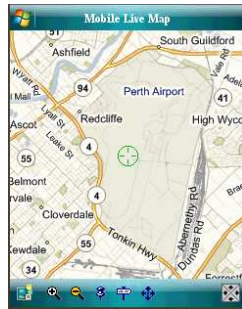
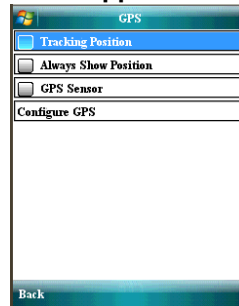
**Find Address**


Find Address

Please input address

China  
United States  
Australia  
Any

Port WA, Australia

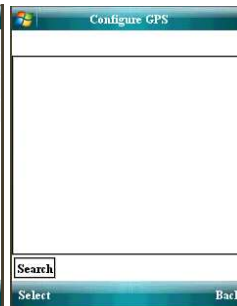
**GPS support**


GPS

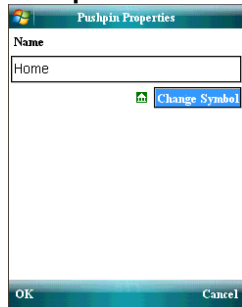
☐ Tracking Position

☐ Always Show Position

☐ GPS Sensor



Configure GPS

**Pushpin**


Pushpin Properties

Name

**System Info**

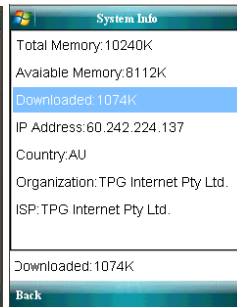

About

Pocket Streets

[www.guidebee.biz](http://www.guidebee.biz)

[www.pstreets.com](http://www.pstreets.com)

Please don't use when driving, this is a freeware, you are free to use, distribute, copy, but the software itself is copyrighted, you are not allowed to decompile, resell etc. without written permission from the original author James Chen. If you want to provide



System Info

Total Memory: 10240K

Available Memory: 8112K

Downloaded: 1074K

IP Address: 60.242.224.137

Country: AU

Organization: TPG Internet Pty Ltd.

ISP: TPG Internet Pty Ltd.

Downloaded: 1074K

This demo provides similar functionality to Microsoft's pocket street. Provide Map Zoom in/Zoom, Pan direction, Find address, add your own pushpin etc.

You can add GPS and navigation yourself to make it navigation software like Tomtom, Navman. I'll leave it to your capable hands☺

## 9.0 Licence

Guidebee Map API provides classes to access map services from Yahoo, Microsoft etc. before using their map service, please read their User Agreement. Your violation of their TOU is at your own risk.

Licence from Guidebee Biz. Includes two parts:

- The licence key

Gives the product name “DigitalMap” and licence keys, six long integers. To correctly use this library, you need to call LicenceManager to add correctly licence to the LicenceManager.

- And the licence file.

The file name is guidebee.lic, which need to put in the root directory of the resource with the MIDlet jar file.

If develops with Wireless tool kit, the licence file needs to put in the res directory. If develops with Netbean, the licence file needs to put in the src directory.

```
LicenceManager licenceManager = LicenceManager.getInstance();
//licence expires on Fri Jun 05 21:56:21 WST 2009
long keys[] = {-0x798e29f774aa73e7L, -0x16ad80bb2211a4d6L,
0x2dbbc31235569af5L, -0x716f5c769512a42bL, 0x1fb1a7d3049565b0L,
0x7d6dada3738dfb43L,};
licenceManager.addLicence("DigitalMap", keys);
```

LicenceManager has two static methods.

```
public static LicenceManager getInstance(MIDlet midlet)
public void addLicence(String appName,long[]keys)
        throws InvalidLicenceException
```

getInstance returns an instance of LicenceManager ,it takes the MIDlet instance which uses the library as the input parameter.

addLicence add licence to the LicenceManager, it takes the appName ,in this case “MapServer” and the licence key.

If the licence key is invalid ,missing or the guidebee.lic is invalid,missing or placed in wrong place, InvalidLicenceException will be thrown, or you forget to add the code to add the licence to the LicenceManager

In this case, you cannot use most of the core API of Map API library.



Licence is sold as one year ,two year and three years period. Licence is only required for developer or company who make use Guidebee Map API to develop their own mobile application. Customers of those mobile application don't need to buy licence for Map APIs.

For detail information, please contact James Shen at [james.shen@guidebee.biz](mailto:james.shen@guidebee.biz)

