University map using java.

**SUBMITTED BY:**

**UTKARSH MAHAJAN 17301973911(ECM)**

**STUTI CHOPRA- 17301973913(ECM)**

**SUBMITTED T0:**

**Er. SHIVINDER DEVRA**

**ABOUT THE PROJECT:**

The project is a **UNIVERSITY** **MAP LOCATOR.**

The map is firstly a general world map which tries to track down any location on the map. It can be examined in 3 different views:

1. STREET VIEW

2. VIRTUAL EARTH

3. SATELLIETE VIEW

Waypoints are put on the desired coordinates. We can add or delete any waypoint at any time. Waypoints of famous locations in the university has been tried to add on the map. On clicking the drop pin on the map, the user will be able to see the site, the picture of the area will pop up with a brief description.

**SUPPORTED OPERATING SYSTEM:**

* **WINDOWS:** This project can easily be configured on windows operating system. For running this project on Windows System you need WAMP or XAMP on your system.
* **Linux:** We can run this project also on all versions of Linux Operating System.
* **Mac:** We can also easily configured this project on Mac operating system

**Tools required:**

1. Apache Netbeans.
2. JXMapViewer jar file.
3. OOPs concepts.
4. Java Swing Knowledge.

**TECHNOLOGY USED:**

THE PROJECT IS DEVELOPED USING THE BELOW TECHNOLOGY:

* **JAVA SWING**: All the validation task and animations has been implemented using Java Swing.
* **JXMAPVIEWER :** JXMAPVIEWER provides a world map that users can directly manipulate to pan and zoom.
* **APACHE NETBEANS:** Layout is designed in NETBEANS.

**STEPS OF DEVELOPMENT:**

1. CREATE PROJECT
2. ADD LIBRARY
3. CREATE MAP WITH JFRAME
4. CREATE EVENT MOUSE
5. CREATE MAPTILE
6. MODIFY FRAME
7. ADD IMAGE
8. CREATE BUTTON WAYPOINT
9. CREATE MY WAYPOINT
10. CREATE WAYPOINT RENDER
11. APPLY WAYPOINT TO MAP
12. CREATE WAYPOINT EVENT.

**WORKING:**

We start by constructing JFrame and add jar folders from library.

SOURCE PACKAGE

|

NEW JAVA PACKAGE (test)

|

NEW JFrame (Main)

When you build a Java application project that has a main class, the IDE automatically copies all of the JAR files on the projects classpath to your projects dist/lib folder. The IDE also adds each of the JAR files to the Class-Path element in the application JAR files manifest file (MANIFEST.MF)

If two JAR files on the project classpath have the same name, only the first JAR file is copied to the lib folder.

\* Only JAR files are copied to the lib folder.

If the classpath contains other types of files or folders, these files (folders) are not copied.

\* If a library on the projects classpath also has a Class-Path element specified in the manifest, the content of the Class-Path element has to be on the projects runtime path.

**SOME OF THE CLASSES USED ARE:**

**DefaultTileFactory :** A tile factory which configures itself using a TileFactoryInfo object and uses a Google Maps like mercator projection.

**VirtualEarthTileFactory:** This command will create both the image tiles and sample HTML file for displaying data with Bing Maps interface.

**MouseInputListeners:** Notified when you change the state of the mouse, or whenever you drag or move the move.

## OSMTileFactoryInfo: Uses Open Street Map

## PanMouseInputListener: Used to pan using press and drag mouse gestures.

## ZoomMouseWheelListenerCenter: Zooms using the mouse wheel on the view center.

## Tile Factory: A class that can produce tiles and convert coordinates to pixels.

## GeoPosition: An immutable coordinate in the real (geographic) world, composed of a latitude and a longitude.

## Interface Waypoint: A Waypoint is a GeoPosition that can be drawn on a may using a WaypointPainter.

## Class WaypointPainter<W extends [Waypoint](https://jar-download.com/javaDoc/org.jxmapviewer/jxmapviewer2/2.4/org/jxmapviewer/viewer/Waypoint.html)>: Paints waypoints on the JXMapViewer. This is an instance of Painter that only can draw on to JXMapViewers.

## Interface WaypointRenderer<W>: A interface that draws waypoints. Implementations of WaypointRenderer can be set on a WayPointPainter to draw waypoints on a JXMapViewer

## Class JXMap Kit: The JXMapKit is a pair of JXMapViewers preconfigured to be easy to use with common features built in. This includes zoom buttons, a zoom slider, and a mini-map in the lower right corner showing an overview of the map. Each feature can be turned off using an appropriate is*X*visible property. For example, to turn off the minimap call. The JXMapViewer is preconfigured to connect to maps.swinglabs.org which serves up global satellite imagery from NASA's [Blue Marble NG](http://earthobservatory.nasa.gov/Newsroom/BlueMarble/) image collection.

## Enum JXMapKit.DefaultProviders: Returns the enum constant of this type with the specified name.

## Class JXMapViewer: A tile oriented map component that can easily be used with tile sources on the web like Google and Yahoo maps, satellite data such as NASA imagery, and also with file based sources like pre-processed NASA images. A known map provider can be used with the SLMapServerInfo, which will connect to a 2km resolution version of NASA's Blue Marble Next Generation imagery. @see SLMapServerInfo for more information. Note, the JXMapViewer has three center point properties. The addressLocation property represents an abstract center of the map. This would usually be something like the first item in a search result. It is a [GeoPosition](https://jar-download.com/javaDoc/org.jxmapviewer/jxmapviewer2/2.4/org/jxmapviewer/viewer/GeoPosition.html" \o "class in org.jxmapviewer.viewer). The centerPosition property represents the current center point of the map. If the user pans the map then the centerPosition point will change but the addressLocation will not. Calling recenterToAddressLocation() will move the map back to that center address. The center property represents the same point as the centerPosition property, but as a Point2D in pixel space instead of a GeoPosition in lat/long space. Note that the center property is a Point2D in the entire world bitmap, not in the portion of the map currently visible. You can use the getViewportBounds() method to find the portion of the map currently visible and adjust your calculations accordingly. Changing the center property will change the centerPosition property and vice versa. All three properties are bound.

## Class DefaultWaypointRenderer: This is a standard waypoint renderer.

**Class DefaultWaypoint:** implements [Waypoint](https://jar-download.com/javaDoc/org.jxmapviewer/jxmapviewer2/2.4/org/jxmapviewer/viewer/Waypoint.html).

## Class AbstractPainter<T>: A convenient base class from which concrete [Painter](https://jar-download.com/javaDoc/org.jxmapviewer/jxmapviewer2/2.4/org/jxmapviewer/painter/Painter.html) implementations may extend. It extends [AbstractBean](https://jar-download.com/javaDoc/org.jxmapviewer/jxmapviewer2/2.4/org/jxmapviewer/beans/AbstractBean.html" \o "class in org.jxmapviewer.beans) as a convenience for adding property change notification support. In addition, AbstractPainter provides subclasses with the ability to cacheable painting operations, configure the drawing surface with common settings (such as antialiasing and interpolation), and toggle whether a subclass paints or not via the visibility property. Subclasses of AbstractPainter generally need only override the [doPaint(Graphics2D, Object, int, int)](https://jar-download.com/javaDoc/org.jxmapviewer/jxmapviewer2/2.4/org/jxmapviewer/painter/AbstractPainter.html" \l "doPaint-java.awt.Graphics2D-T-int-int-) method. If a subclass requires more control over whether caching is enabled, or for configuring the graphics state, then it may override the appropriate protected methods to interpose its own behavior.

## A tile oriented map component that can easily be used with tile sourceson the web like Google and Yahoo maps, satellite data such as NASA imagery,and also with file based sources like pre-processed NASA images.

## Note, the JXMapViewer has three center point properties. The <B>addressLocation</B> property represents an abstract center of the map. This would usually be something like the first item in a search result. It is a {@link GeoPosition}. The <b>centerPosition</b> property represents the current center point of the map. If the user pans the map then the centerPosition point will change but the <B>addressLocation</B> will not. Calling <B>recenterToAddressLocation()</B> will move the map back to that center address. The <B>center</B> property represents the same point as the centerPosition property, but as a Point2D in pixel space instead of a GeoPosition in lat/long space. Note that the center property is a Point2D in the entire world bitmap, not in the portion of the map currently visible. You can use the <B>getViewportBounds()</B> method to find the portion of the map currently visible and adjust your calculations accordingly. Changing the <B>center</B> property will change the <B>centerPosition</B>property and vice versa. All three properties are bound.

## USE: Generally, we use maps as a reference to show political boundaries, landforms, water bodies, and the positions of cities. Maps also help us to know the routes of an area, landmarks, a building or things, etc.

## As this map is customized for Guru Nanak Dev University, which is spread in across 500acres, thus a map with location from one place to another, with pictures and description of certain places will help freshers’ and outsiders to locate the areas easily.