LogiMap Template Documentation

Introduction

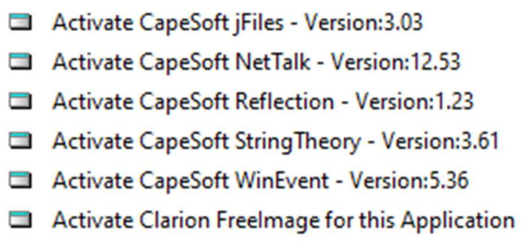
LogiMap Template provides a way to create routes and stops from a large number and dispersion of packets to be delivered in different directions. The template is very technical and needs a good understanding of NetMaps and its classes. As a complement you can access the documentation of the same https://www.capesoft.com/docs/NetTalk11/NetTalkMaps.Htm

LogiMap easily allows to:

- Load addresses and size/weight of packages to be delivered
- Create and memorize distribution areas
- Modify those base areas according to the distribution of addresses
- Automatically detect the addresses belonging to each Area
- Detect orphan addresses, which are outside the Areas
- Create Bird's Eye Flight Route with a quick estimate of time and distance
- Create Routes for each Area
- Optimize those routes according to traffic conditions
- Control maximum volumes and weights to be loaded to trucks
- And many more features to help you program your logistics system

Installation

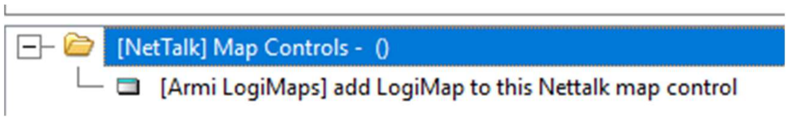
An installer is provided, run it by choosing the Clarion directory where to install it.
Required templates for NetMaps:


Activate CapeSoft jFiles - Version:3.03
Activate CapeSoft NetTalk - Version:12.53
Activate CapeSoft Reflection - Version:1.23
Activate CapeSoft StringTheory - Version:3.61
Activate CapeSoft WinEvent - Version:5.36
Activate Clarion FreeImage for this Application

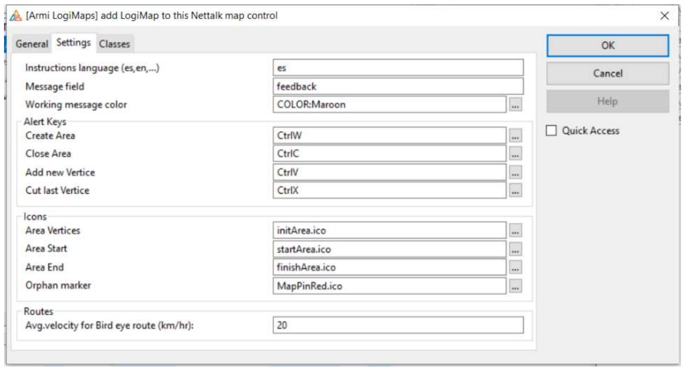
Log in to Clarion and add the LogiMaps template to the Registry.
Examples are provided in ClarionXX/accesory/Armi/Logimaps for Clarion10 and 11 ABC.
The language is defined by equates, they are defined in the file ArmiLogiMaps.trn found in ClarionXX/LibSrc/win, if you need English comment the lines in Spanish and uncomment the lines in English.
Compile, if everything is correct, click on the "Markers" button and you will see the first approach to LogiMap seeing some Areas already drawn on the map.

Template settings

In the demo this is all done but if you need to add the template to a new app, add the global extension to the app and then, in the window where you have the NetMaps object, separate it and press Insert to hang the LogiMap template from it choosing it from the list like any other Clarion template. It will look like this:


[NetTalk] Map Controls - ()
[Armi LogiMaps] add LogiMap to this Nettalk map control

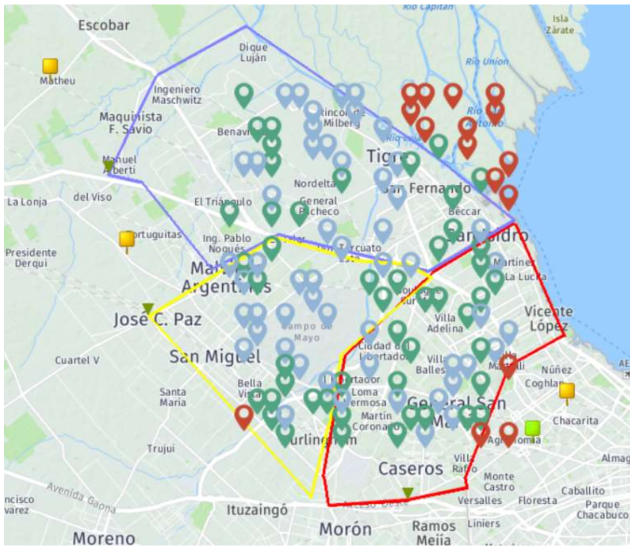
In this last version you need to set NetMaps to use the NetMapsHerePlatform class


[Armi LogiMaps] add LogiMap to this Nettalk map control

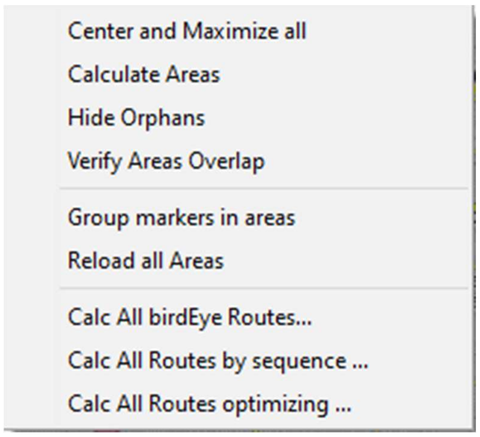
Instructions language (es,en,...)	es
Message field	feedback
Working message color	COLOR:Maroon
Alert Keys	
Create Area	CtrlW
Close Area	CtrlC
Add new Vertice	CtrlV
Cut last Vertice	CtrlX
Icons	
Area Vertices	initArea.ico
Area Start	startArea.ico
Area End	finishArea.ico
Orphan marker	MapPinRed.ico
Routes	
Avg.velocity for Bird eye route (km/hr):	20

You can modify alert keys, icons, etc. The suggested icons are provided in the example images directory, simply copy them to your app's directory.
To create an Area, press "Ctrl-W", the cursor will become a crosshair, move the cross to where you want to start the area and press "Ctrl-V" to create at the starting point, continue moving the cursor to the other vertices and creating them with "Ctrl-V", to finish press "Ctrl-C" and automatically close the area. If you make a mistake with one you can press "Ctrl-X" to delete it.
Now press the "Random Markers" button at the top right, this button randomly simulates the creation of your delivery points. Studying its code you will have an idea of how to add your deliveries to the map.

Assuming the following random distribution of points:



By now pressing the right mouse button on the map you will get the following functions:



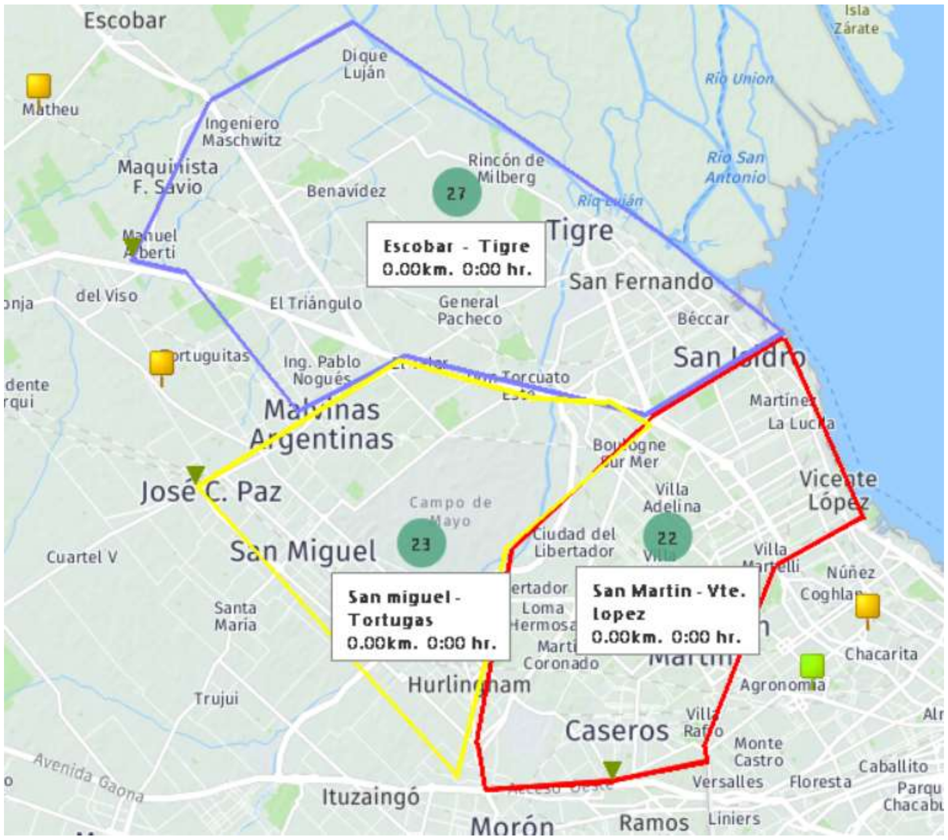
Center and maximize all: Center the map by zooming in to show all the objects on it.

Calculate Areas: Detects which delivery point belongs to each area, as well as orphaned points (that are out of area) and paints them red.

Hide Orphans: Hide orphaned points

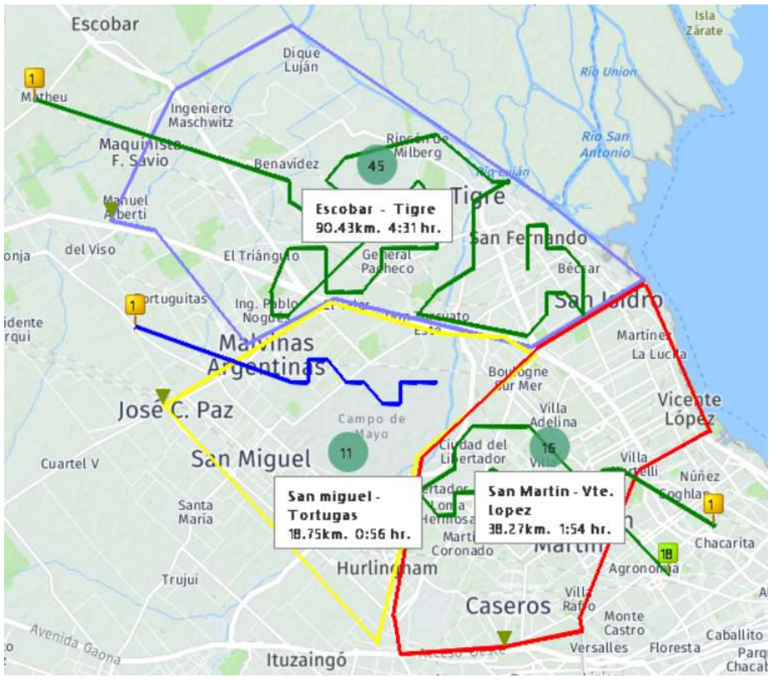
Verify Areas Overlap: verifies that no area is drawn occupying the place of another, and indicates in which vertice the problem is to solve it.

Group markers in Areas: Deletes the points and indicates in a central circle the number of points within each area.



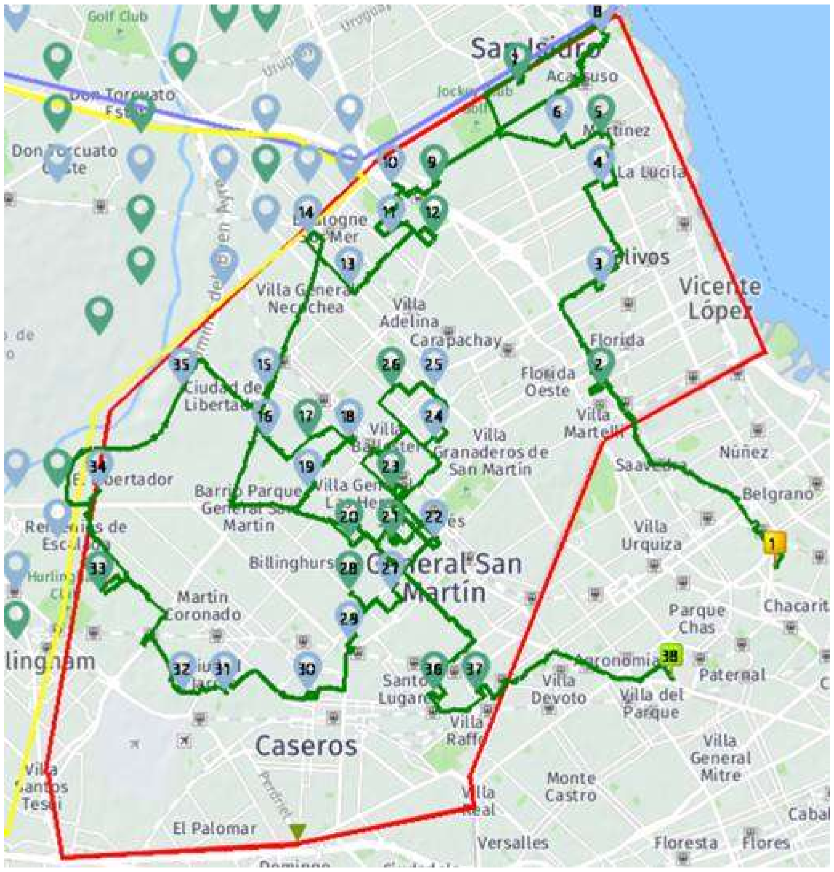
Reload all Areas: Displays the points again.

Calculate all Bird Eye Routes: Calculate the routes in each area as if it were flying without taking into account the streets. If you now pressed Group stops in areas you would have a general screenshot of stops, kilometers and times in each area.

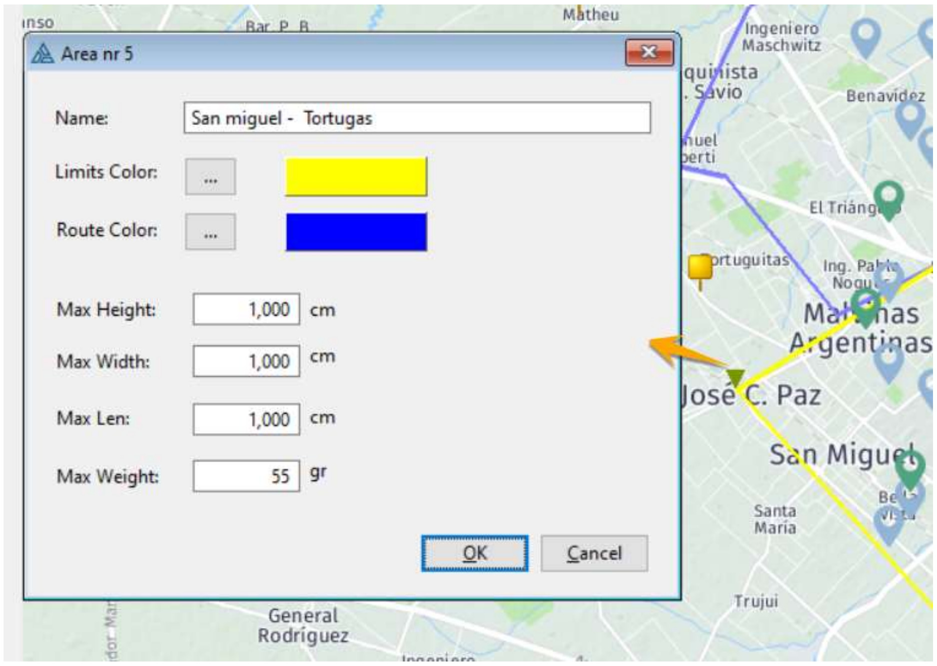


Calculate all routes by sequence: Calculate routes by going on roads using the sequence of stops indicated by you when loading the stops.

Calculate all routes optimizing: Calculate the routes by roads but optimizing the route by creating a new sequence of stops.



Each Area data includes the maximum capacity data of the truck that will be assigned to it to make the delivery, being able to calculate the maximum according to dimensions and weight of the sum of the packages to be delivered, by clicking on the icon of each area:



Area nr 5

Name: San miguel - Tortugas

Limits Color: ...

Route Color: ...

Max Height: 1,000 cm

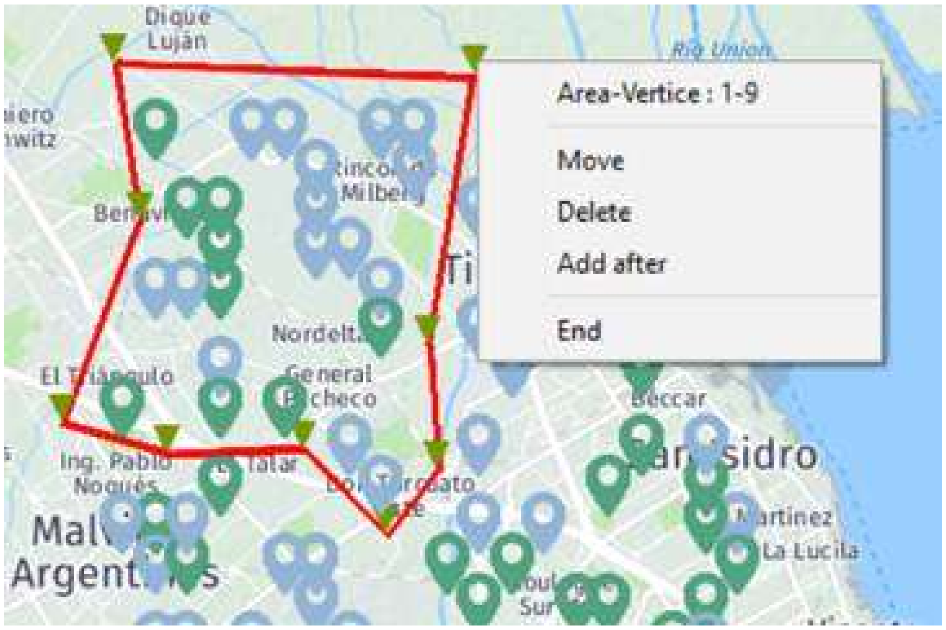
Max Width: 1,000 cm

Max Len: 1,000 cm

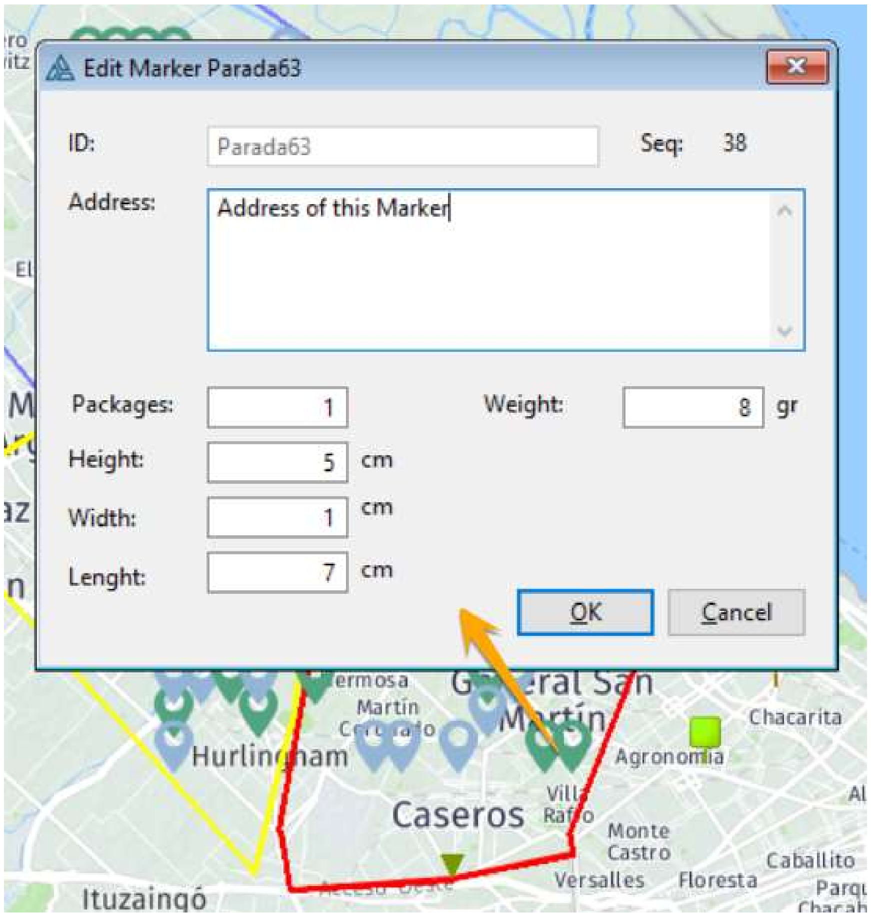
Max Weight: 55 gr

You can modify and adapt each area to include or not orphans, being able to move or delete any of its vertices.

To modify the right-click area on the green triangle that indicates the beginning of the area, that will make the same icon appear in each vertice of the area and you can, using also the right button on each vertice, move it, delete it, etc.



Each address can be viewed and modified:


Edit Marker Parada63

ID: Parada63 Seq: 38

Address: Address of this Marker

Packages: 1 Weight: 8 gr

Height: 5 cm

Width: 1 cm

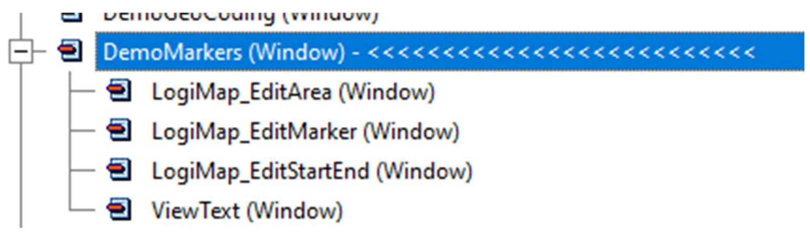
Lenght: 7 cm

OK Cancel

Finally, all information is handled in easily accessible memory queues to be recorded or worked on in your logistics system.

By pressing the "Inspect" button, at the top right you can see each queue generated by the template from where to obtain information to take to your logistics or management system.

In the installation, the template adds some windows with which the editing is done.


DemoGeoCoding (Window)
DemoMarkers (Window) - <<<<<<<<<<<<<<<<<<<<<<<<<<
LogiMap_EditArea (Window)
LogiMap_EditMarker (Window)
LogiMap_EditStartEnd (Window)
ViewText (Window)

The template is provided with source code.

Class, methods, and parameters

Everything is delivered with source code so you have .inc and .clw to investigate the class, we now show the most outstanding.

Queues, definition:

Areas, supports up to 64 vertices each.

```
AreasQueueType      Queue,TYPE
Nr                      long
Name                    string(101)
Color                   long
GUID                    string(16)
Vertices                long, dim(64)
VerticesCount           long
Stops                   long
Distance                LONG
Time                    LONG
VelocityAvg             LONG
RouteColor              LONG
HasStart                LONG
HasEnd                  LONG
                    End
```

Vertices:

```
VerticesQueueType    QUEUE,TYPE
AreaNr                  long
Nr                      long
REAL  Latitude
ACTUAL  length
                    END
```

Start and end points of each Area:

```
StartEndQueueType   QUEUE,TYPE
AreaNr                  long
Type                    string(10)
Name                    string(255)
REAL  Latitude
ACTUAL  length
                    END
```

Area route points:

```
StopsQueueType      QUEUE,TYPE
AreaNr                  long
MarkerID                string(255)
REAL  Latitude
ACTUAL  length
Sequence                long
Distance2Start          long
Distance2previous           LONG
DistanceRunning         long
Time2next               long
TimeRunning             long
                    END
```

Route instructions:

```
InstructionsQueueType       QUEUE,TYPE
AreaNr                          LONG
LegNumber                       LONG
Time                            LONG
Distance                        LONG
InstructionText                 STRING(1024)
                            END
```

```
AreasQueue        Definition of each Area
StartEndQueue     Each point of Start and End of each Area
StopsQueue        Stops of each route of each area
OrphansQueue      Orphan points
RouteQueue        (NetTalk) Calculated routes
WaypointsQueue    (NetTalk) Each delivery point
ManeuverQueue     (NetTalk) Intermediate points of each route
InstructionsQueue (NetTalk) Colloquial instructions on what to do
```

Main methods and parameters:

DeleteAllAreas procedure(long pRefresh=1)**Deletes all Areas, their vertices and the Path, example:DeleteAllAreas(**
0) clears the window without refreshing

SetArea procedure(long pNr,string **pName**,<long pColor>,<long **pRouteColor**>,<string pGuid>) Creates an Area, with its name and color, GUID can be used to indicate the GUID or ID of your management system, example:SetArea(1

,'**North West**',16744448,32896)

SetAreaVertice **procedure**(long a, **long** in, **real** pLatitude, **real** pLongitude)
Create a vertice of an Area, example:
SetAreaVertice(1,1,-33.8851739726444,   18.5310778623994)

SetStartEnd procedure(long **pAreaNr,string** pType,real pLatitude, **real** pLongitude)Creates a route start or end point of an Area, the pType can be just 'Start' or 'End', the start and end icons are defined with the following parameters:StartIcon string

(255)
EndIcon   **string**(255)
example:
SetStartEnd(1,'Start',-33.8638843,18.5103867)

DrawAllAreas procedure(long pRefresh=1)**Redraws all areas, example:**DrawAllAreas
(0
)SetDelivery **Procedure**

(String pId, **Real** pLatitude, Real **pLongitude**, String **pIcon** , long pSequence, long pPackages=0, **long pHeight=0**, **long pWidth=0**, **long pLength**=0,   long pWeight=0, Long pIconIndex=1, **Long pPointX**=16, Long pPointY = 32   ,   Long pHide=false)
Creates a delivery point, this method basically does the same as the **NetTalk SetMarker()** but can also keep additional point data that is not foreseen in NetTalk, example:
SetDelivery('Stop'&m#,'Stop
address'&m#,Lat$,Lon$,**choose**(**int**(m#/2)=m#/2,'**MapPinLightBlue.ico**','MapPinGreen.ico'),
0,1,**random**(1,9),**random**(1,9),**random**(1,9),**random**(1,9))

Example of creating Areas by code:

```
lm.DeleteAllAreas(0)
lm.HideOrphans = 0
lm.ShowOnlyTotalMarkers = 0

lm.SetArea(1,'North West',16744448,32896)
lm.SetAreaVertice(1,1,-33.8851739726444,       18.5310778623994)
lm.SetAreaVertice(1,2,-33.8868842364711,       18.524205641905)
lm.SetAreaVertice(1,3,-33.9139634137285,       18.4949987048042)
lm.SetAreaVertice(1,4,-33.9179540293243,       18.4781617645932)
lm.SetAreaVertice(1,5,-33.9213745569779,       18.4383028857262)
lm.SetAreaVertice(1,6,-33.9236549087469,       18.4321178872813)
lm.SetAreaVertice(1,7,-33.9310660519963,       18.4369284416273)
lm.SetAreaVertice(1,8,-33.9484537342352,       18.4654481566787)
lm.SetAreaVertice(1,9,-33.9447481626106,       18.4840031520133)
lm.SetAreaVertice(1,10,-33.9527293938022,      18.4960295378784)
lm.SetAreaVertice(1,11,-33.9507340860043,      18.5163025883366)
lm.SetAreaVertice(1,12,-33.9561499214558,      18.5403553600667)
lm.SetAreaVertice(1,13,-33.9267903924294,      18.5455095254374)
lm.SetAreaVertice(1,14,-33.9125381938729,      18.5410425821161)
lm.SetStartEnd(1,'Start',-33.8638843,18.5103867,'startArea.ico')

lm.DeleteAllRoutes()
lm.DrawAllAreas(0)
lm.MapMaximizeAndCenter(0
lm.FindAllMarkers()
lm.IdentifyOrphanMarkers(1)

lm.HideMapMenu          Disables the map general menú.
lm.HideAreaMenu         Disables the Area menú.
lm.DonNotDraw           Disables the Area Drawing.
lm.SequenceFont         Sets the Font name of the icons sequence numbers
                        By default ‘Arial’
lm.SequenceFontSize     Sets the font size, by default 9
```

Example of creating random delivery points:

```
Latitude = -34.5
Longitude = -58.6
Margen = 10
loop m#=1 to random(50,50)
    Lat$ = Latitude+random(-10,10)/100
    Lon$ = Longitude+random(-Margen,Margen)/100
    net.MarkerQueue.Latitude  = Lat$
    net.MarkerQueue.Longitude = Lon$

    get(net.MarkerQueue,net.MarkerQueue.Latitude,net.MarkerQueue.Longitude)
    if error()
        lm.SetDelivery('Parada'&m#,'Dirección de la parada
'&m#,Lat$,Lon$,choose(int(m#/2)=m#/2,'MapPinLightBlue.ico' ,'Map-
PinGreen.ico'),0,1,random(1,9),random(1,9),random(1,9),random(1,9))
        ! (String pId, String pDescription, Real pLatitude, Real pLongitude, String
pIcon, long pSequence, long pPackages=0, long pHeight=0, long pWidth=0, long
pLength=0, long pWeight=0, Long pIconIndex=1, Long pPointX=16, Long pPointY = 32,
Long pHide=false )
    END
END
```

End of document