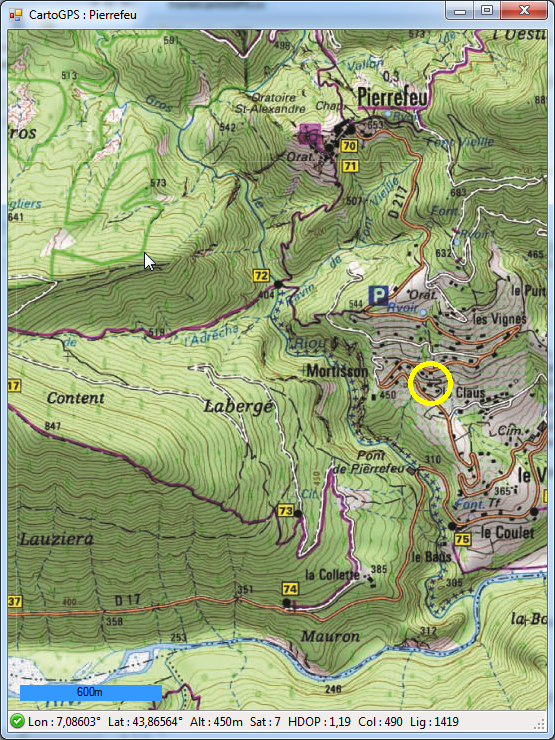
**CartoGPSX**

**J. Lemaire**

Pierrefeu – Février 2020

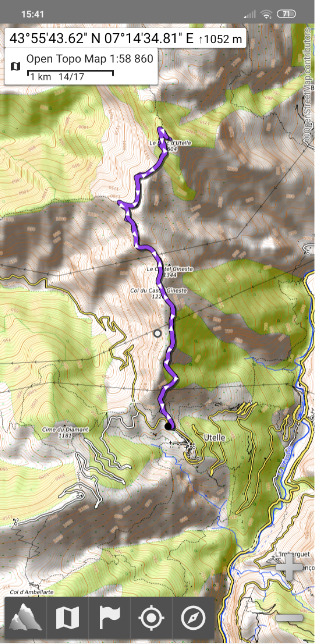
# Objectifs

Il y a quelques années, une charmante galère « hors sentiers » dans les ubacs du Cheiron et de Conségudes, nous a incité à réaliser une petite application Windows, pour une tablette tactile avec un capteur GNSS[[1]](#footnote-2) sur son port USB ; son but était d’afficher, sur une carte pré-chargée, la position courante indiquée par le capteur, de manière à pouvoir se situer par rapport au sentier à suivre :



Cependant, même si cette application s’est révélée utile pour la randonnée et en particulier pour rechercher des sites hors sentiers (par exemple l’Infernet Aven, près des Ferres), elle restait un peu fastidieuse à mettre en œuvre, car il fallait au préalable y charger manuellement la carte adéquate et la caler[[2]](#footnote-3), en utilisant par exemple le site Géoportail[[3]](#footnote-4).

Or récemment, une application de randonnée sous Android, **AlpineQuest**[[4]](#footnote-5), nous a montré qu’un smarphone, disposant d’un capteur GNSS intégré, pouvait offrir ce type de fonctionnalité, avec un chargement automatique des portions de cartes adéquates, calées et à différentes échelles (on parle alors de tuiles) ; on peut par exemple accéder ici aux cartes **Open Topo Map**[[5]](#footnote-6) :



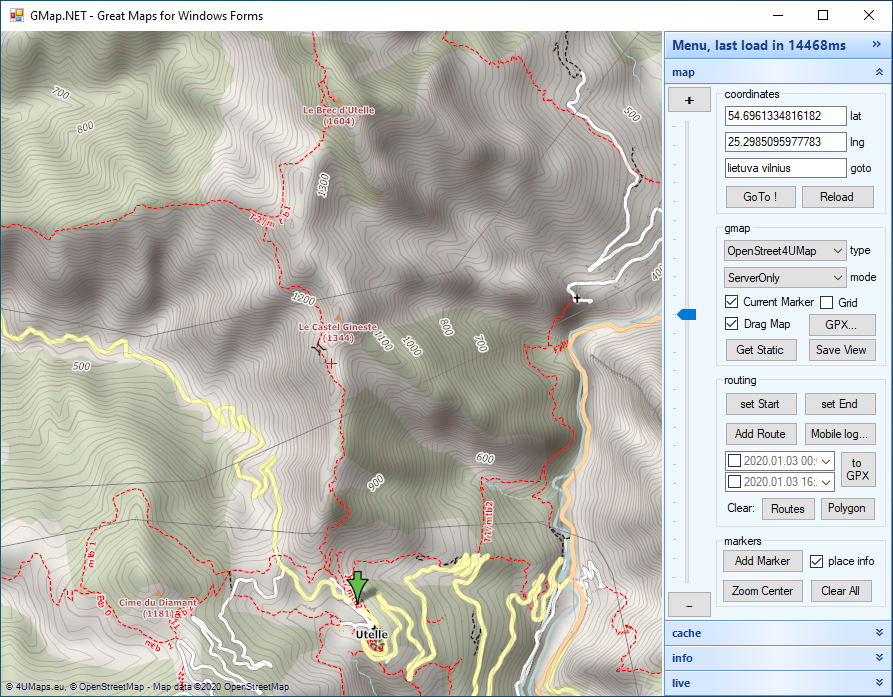
Avec la version payante de cette application[[6]](#footnote-7), on peut même stocker ces portions de cartes pour que celles-ci soit ensuite disponibles hors connexion.

**AlpineQuest** permet également de visualiser des **traces GPX**[[7]](#footnote-8) et, dans sa version payante, d’en créer en utilisant les données fournies en continu par le capteur GNSS.

Fort de cette constatation, il était naturel de voir si tout cela ne pouvait pas être mis à profit dans CartoGPS, de manière à simplifier son utilisation. Et c’est là que nous avons découvert une solution « miracle » existant depuis longtemps : **GMap.NET** !

# GMap.NET[[8]](#footnote-9) [[9]](#footnote-10)

**GMap.NET** est le nom d’un contrôle utilisateur .NET, développé par Radioman.it, en Lituanie, précisément destiné à l’affichage de tuiles cartographiques, avec éventuellement en superposition, des marqueurs, des traces GPX, etc.



Les tuiles peuvent être choisies dans un vaste ensemble de bases de données cartographiques en libre accès chez des « providers ».



De plus, ce contrôle enregistre automatiquement les tuiles affichées dans un cache de 32MB au maximum, de manière à les rendre disponibles ensuite, hors connexion.

Il répond donc exactement et très simplement aux objectifs précédent, si ce n’est que les cartes d’**Open Topo Map** ne sont pas ici proposées !

Pour le mettre en œuvre dans une application Windows Form en C# avec Visual Studio, il suffit de s’inspirer de l’exemple **Demo.WindowsForm**, fourni ici avec le code source des projets **GMap.NET.Core** et **GMap.NET.WindowsForms**. Cependant, comme cet exemple est assez complexe, il n’est pas inutile de commencer par lire quelques aides en lignes :

* GMAP.NET (Independant Software)[[10]](#footnote-11).
* GMap Complete Guide (Balaaji S)[[11]](#footnote-12).

On peut ainsi comprendre la démarche à suivre :

1. Créer une solution avec un projet de démarrage de type Windows Form en C#.
2. Ajouter à cette solution les 2 projets de bibliothèques de classes (.dll) en C#, **GMap.NET.Core** et **GMap.NET.WindowsForms**, avec leurs ressources, téléchargeables à partir du site Github de Radioman[[12]](#footnote-13).
3. Ajouter des références vers ces 2 projets dans le projet de démarrage. A noter qu’on peut également faire cela avec le gestionnaire de paquets NuGet, mais sans qu’on puisse ensuite modifier le code des 2 bibliothèques de classes, comme nous allons le faire.
4. Dans la fenêtre principale de l’application :
   1. ajouter les instructions **using** pour utiliser ces classes :

using GMap.NET.WindowsForms;

using GMap.NET.WindowsForms.Markers;

* 1. ajouter un contrôle **GMapControl**
  2. ajouter les instructions suivantes dans l’implémentation de l’événement OnLoad :

<nom contrôle>.MapProvider = GMap.NET.MapProviders.<nom provider>.Instance;

GMap.NET.GMaps.Instance.Mode = GMap.NET.AccessMode.ServerAndCache;

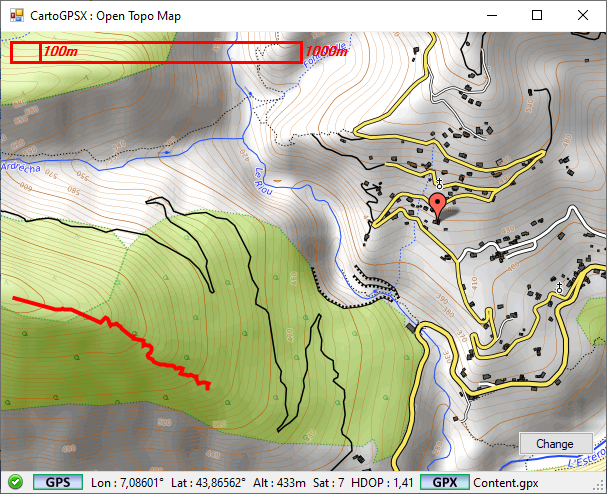
1. Pour afficher des marqueurs et des routes GPX, utiliser un overlay.

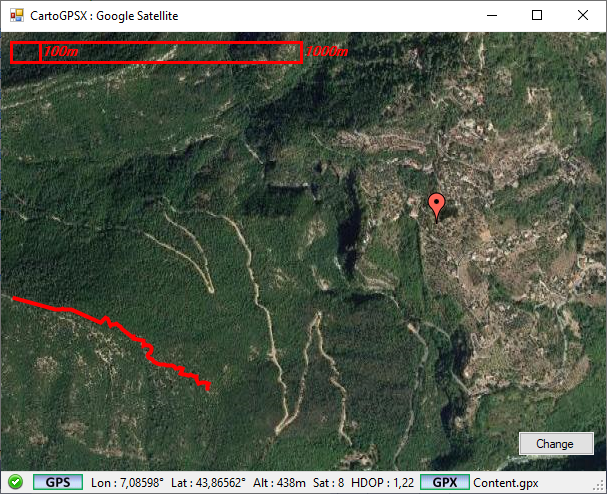
Les principes de cette mise en œuvre sont donc extrêmement simples à comprendre et il suffit de se reporter aux aides précitées pour les détails supplémentaires (zoom, changement de position, affichage ou non des composants de l’overlay, etc.).

# L’application CartoGPSX

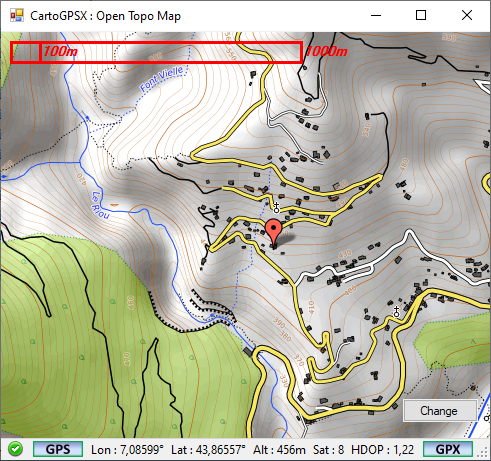
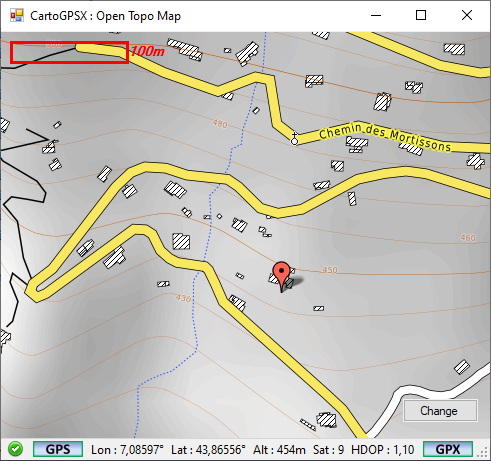
## Présentation

Sur une portion de carte de type **Open Topo Map** ou **Google Satellite**,  cette application Windows de type Windows Form permet d’afficher la position d’un point dont on connaît les coordonnées géographiques (latitude et longitude) ; elle permet également d’afficher une trace GPX, constituée d’une suite de tels points :





La portion de carte doit d’abord être chargée « on line » dans un cache local, en étant donc connecté à Internet : il suffit juste pour cela de l’afficher dans la fenêtre, avec une résolution suffisante. On pourra ensuite y accéder « off line »[[13]](#footnote-14), uniquement en utilisant les données du cache, qui peut contenir plusieurs types de cartes. Par rapport à **CartoGPS**, cette phase de chargement est donc considérablement simplifiée et permet en outre d’afficher une même carte avec différents zooms et ses niveaux de détails adaptés :

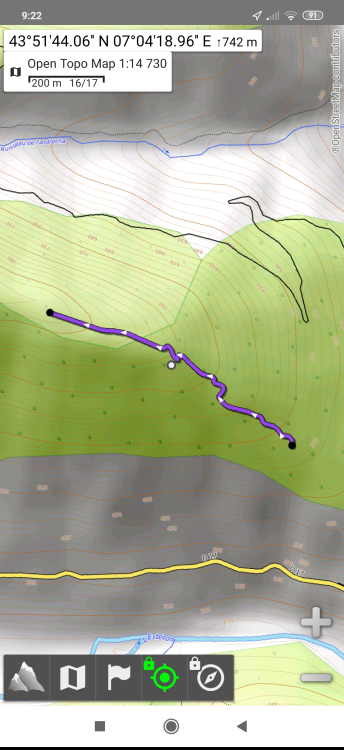
Les coordonnées géographique du point à représenter sont fournies par un capteur GNSS **DP0107** (Drotek)[[14]](#footnote-15), basé sur le module **NEO-M8N** d’Ublox[[15]](#footnote-16) et monté sur une carte disposant d’une antenne, d’une petite pile pour le démarrage « à chaud » et d’un port USB. Avec la possibilité d’exploiter 3 systèmes satellitaires en parallèle (GPS, GLONASS et Galileo par exemple), ce module peut fournir une position géographique avec une précision horizontale <5m. Par rapport à ses prédécesseurs (gamme M5, M6 et M7) la consommation a été réduite ainsi que le temps d’acquisition.

Ce module sera inclus dans un boitier de protection facilement réalisable avec une imprimante 3D :



Les traces GPX peuvent être créées manuellement[[16]](#footnote-17), ou automatiquement avec un enregistreur de trace[[17]](#footnote-18). C’est ce que nous avons fait ici, avec un Xiaomi MI9 SE :



Les fichiers GPX sont de de type XML et principalement constitués d’une succession de points :

<?xml version=**"1.0"** encoding=**"UTF-8"**?>

<gpx xmlns=**"http://www.topografix.com/GPX/1/1"** version=**"1.1"** creator=**"AlpineQuest 2.2.3"** xmlns:xsi=**"http://www.w3.org/2001/XMLSchema-instance"** xsi:schemaLocation=**"http://www.topografix.com/GPX/1/1 http://www.topografix.com/GPX/1/1/gpx.xsd"**>

<metadata>

<desc>**Trace AlpineQuest 26/12/2019 15:35:49**</desc>

<link href=**"https://www.alpinequest.net"** />

<time>**2019-12-26T18:08:41Z**</time>

</metadata>

<trk>

<name>**Content**</name>

<desc>**Total: 1.0km**</desc>

<trkseg>

<trkpt lat=**"43.8604715"** lon=**"7.0761368"**>

<ele>**631.7739868164062**</ele>

<time>**2019-12-26T14:35:49Z**</time>

</trkpt>

<trkpt lat=**"43.8605438"** lon=**"7.0761527"**>

<ele>**624.8319702148438**</ele>

<time>**2019-12-26T14:36:24Z**</time>

</trkpt>

<trkpt lat=**"43.8606159"** lon=**"7.076132"**>

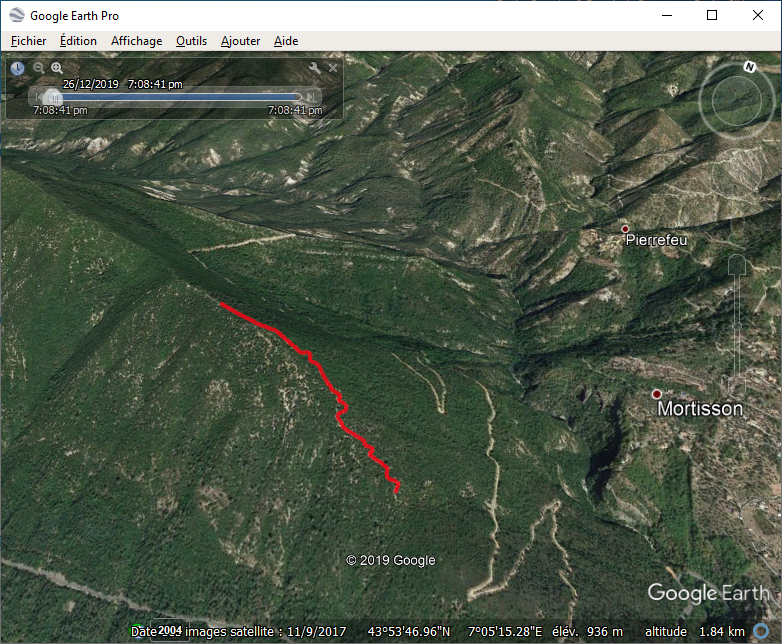
<ele>**634.1160278320312**</ele>

<time>**2019-12-26T14:36:58Z**</time>

</trkpt>

Etc.

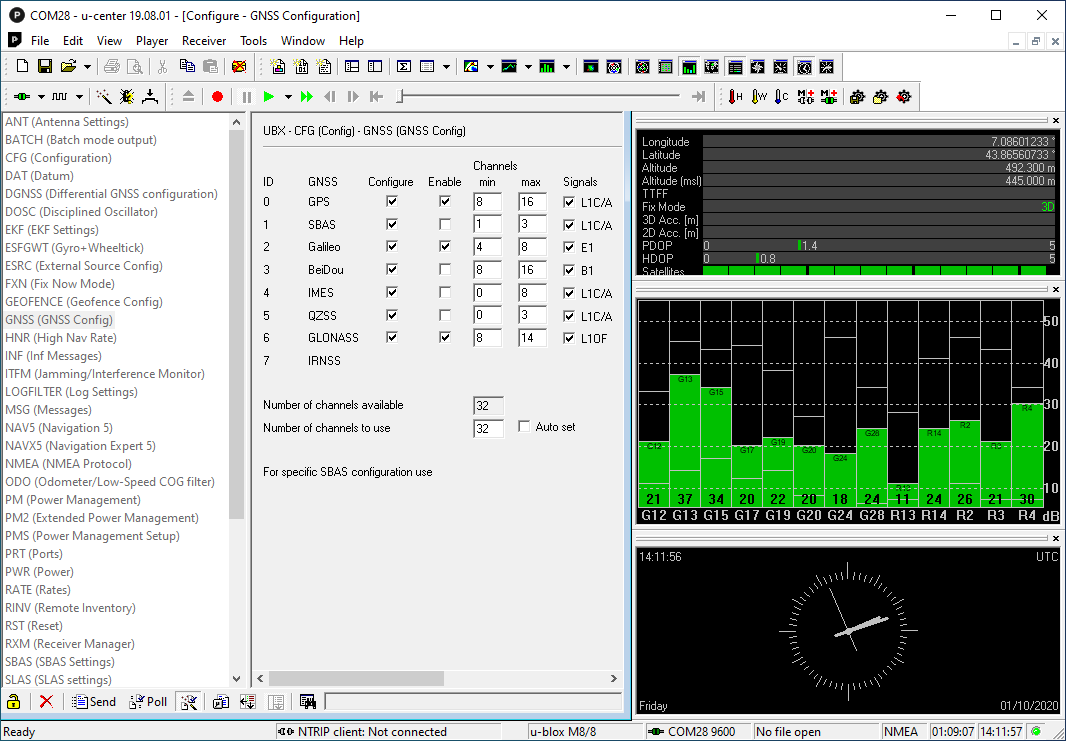
En les transformant en fichiers en fichier KML[[18]](#footnote-19), on peut les afficher clairement dans Google Earth :



## Paramétrage du GNSS avec u-Center

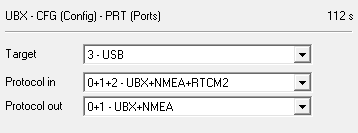
Le logiciel **u-Center**[[19]](#footnote-20) est utilisé ici pour tester le module **NEO-M8N**, le paramétrer et sauvegarder sa configuration, de manière permanente, dans sa mémoire Flash.

Si on veut exploiter simultanément les systèmes GPS, GLONASS et Galileo, il faut ici modifier la configuration par défaut avec une commande **UBX-CFG-GNSS** :

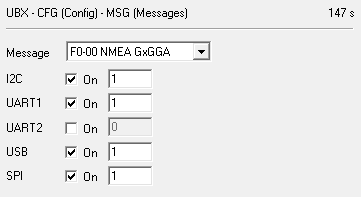


Puis l’enregistrer avec la commande **Receiver/Action/Save Config**.

On notera également que, par défaut, les protocoles UBX et NMEA sont activés sur le port USB (cf. **UBX-CFG-PRT**) :



et qu’on reçoit en particulier les trames textuelles **$GNGGA** sur ce port (cf. **UBX-CFG-MSG**) :



Celles-ci contiennent toute l’information qui sera ici exploitée :

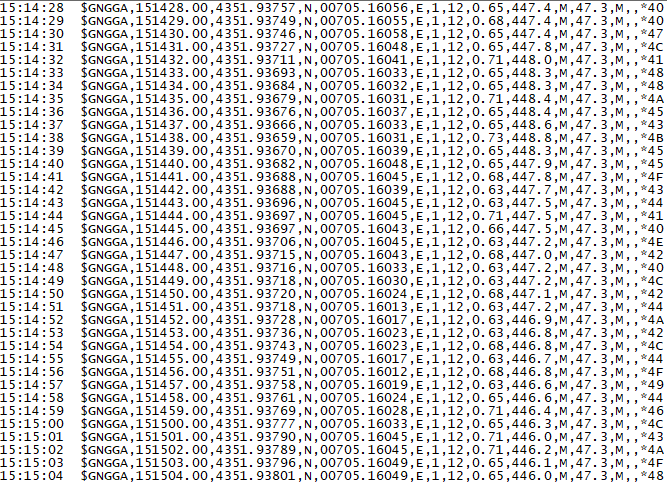


à savoir[[20]](#footnote-21) :

* Latitude (43,5193772° N).
* Longitude (7,0515951° E).
* Nb de satellites (11).
* HDOP (0,81).
* Altitude (457m).

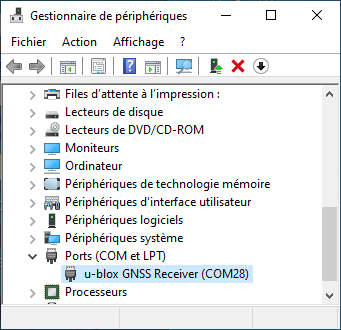
On peut le constater avec la vue « **Text Console** ».

Evidemment, on peut réduire les informations transmises à ces seules trames en sortie sur le port USB avec les commandes **UBX-CFG-PRT** et **UBX-CFG-MSG**, ce que nous ferons donc ici :



## Détection du port série pour le capteur GNSS

Quand on connecte le capteur GNSS **DP0107** à un PC via son port USB, un port série virtuel est ajouté au système ; on peut le constater avec le **Gestionnaire de périphérique** :



Son nom complet, **u-blox GNSS Receiver (COM28)**, permet de l’identifier et fournit le numéro du port série à questionner pour récupérer les informations transmises par le capteur.

Il existe un moyen très simple pour obtenir ce numéro dans une application Windows Form en C# : utiliser la bibliothèque **ORMi**[[21]](#footnote-22) qui permet d’accéder aux ressources du système, via l’interface WMI[[22]](#footnote-23) et avec des requêtes SQL[[23]](#footnote-24).

Après avoir ajouté une référence vers **ORMi** avec le gestionnaire de packages NuGet, il suffit ici de définir une classe Device :

**[**WMIClass**(**"Win32\_PnPEntity"**)]**

**public** class Device

**{**

// Pour GetPortName()

**public** string Name **{** **get;** **set;** **}**

**}**

et de lancer une requête de recherche du pilote de périphérique contenant « u-bloc GNSS Receiver » dans son nom :

**private** String GetPortName**()**

**{**

String str **=** ""**;**

WMIHelper helper **=** **new** WMIHelper**(**"root\\CimV2"**);**

Device device **=** helper**.**Query**<**Device**>().**ToList**()**

**.**Where**(**p **=>** **(**p**.**Name **??** ""**)**

**.**Contains**(**"u-blox GNSS Receiver"**)).**SingleOrDefault**();**

**if** **(**device **!=** **null)**

**{**

str **=** device**.**Name**;**

int i **=** str**.**IndexOf**(**"("**);**

int j **=** str**.**LastIndexOf**(**")"**);**

str **=** str**.**Substring**(**i **+** 1**,** j **-** i **-** 1**);**

**}**

**return** str**;**

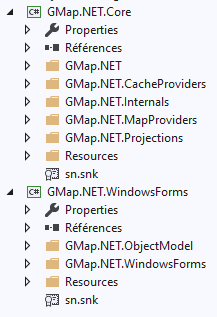
**}**

**Cette méthode sera appelée lors du lancement du programme (méthode de traitement de l’événement Load) et il faudra donc que le capteur GNSS soit connecté à ce moment là.** Ensuite, on pourra le déconnecter puis le reconnecter sans problème car le numéro du port sera maintenu.

## Solution CartoGPSX.sln dans Visual Studio

Dans Visual Studio (version Community 2019), on commence par créer un projet, nommé **CartoGpsx**, de développement d’une application Windows Form en C#, avec **FormCartoGpsx.cs** comme fichier de code pour la fenêtre principale.

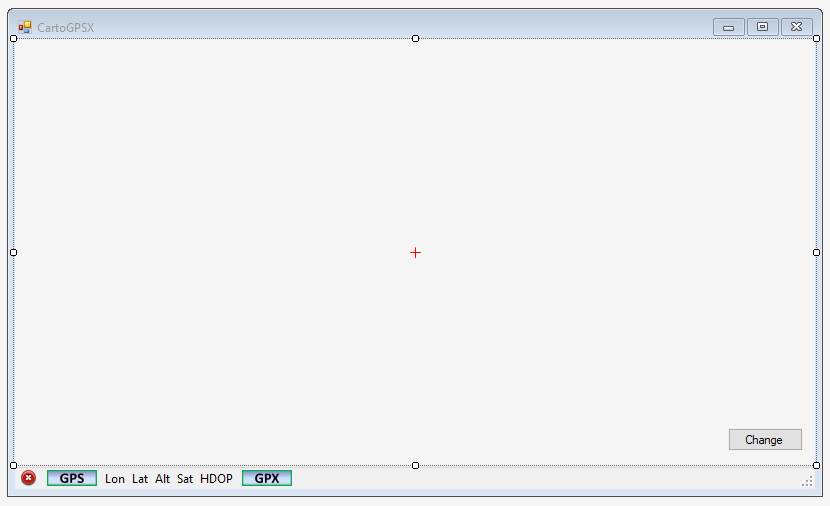
Comme indiqué précédemment, on ajoutera les projets **GMap.NET.Core** et **GMap.NET.WindowsForms**, avec les composants suivants :



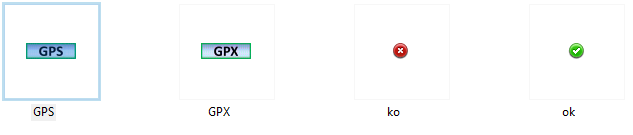
Comme indiqué précédemment, dans le projet **CartoGpsx**, on ajoutera une référence vers la bibliothèque **ORMi**[[24]](#footnote-25), en utilisant le gestionnaire de package NuGet. On a choisi ici la dernière version sans dépendance de ce package : 2.5.0. Par contre, comme sa cible est le Framework .NET 4.6, il faudra éventuellement modifier ce champ dans les propriétés du projet :



Toujours dans le projet **CartoGpsx**, on ajoutera des références vers les 2 projets **GMap.NET.Core** et **GMap.NET.WindowsForms**. Ceci permettra d’ajouter un contrôle **GMap.NET** (nommé **gmap**) dans la fenêtre principale, avec ancrage sur les bords ; il occupera toute la partie de la fenêtre située au-dessus d’un bandeau d’état, **sst**, instance de **StatusStrip**, avec plusieurs champs instances de **ToolStripStatusLabel**, **tsslIco**, **tsslGps**, …, **tsslGpx** :

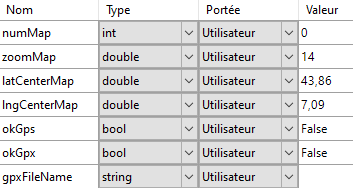


Les 2 premiers et le derniers afficheront une image choisie parmi les ressources du projet, ajoutées précédemment dans un fichier **Resource.resx** :



Comme indiqué dans le design de cette fenêtre principale, on ajoutera également un bouton, **btnChangeMap**, ancré dans son coin inférieur droit et un objet **port**, instance de **SerialPort**

On définira enfin les paramètres suivants dans un fichier **Set.settings** :



Pour implémenter le code C# de la fenêtre principale dans le fichier **FormCartoGpsx.cs**, on commencera par ajouter des méthode de traitement des événement suivants :

* Load : **FormCartoGpsx\_Load.**
* FormClosing : **FormCartoGpsx\_FormClosing.**
* Click : **TsslGps\_Click.**
* Click : **TsslGpx\_Click.**
* Click : **BtnChangeMap\_Click.**
* DataReceived : **Port\_DataReceived.**

Ensuite, dans la classe FormCartoGpsx, on effectuera les ajouts suivants :

* Champs privés : **numMap**, **okGps**, **okGpx**, **gpxFileName**.
* Méthode **FormCartoGpsx\_Load** : ouverture port série (si possible) et initialisation de gmap.
* Méthode **BtnChangeMap\_Click** pour changer ce type.
* Méthode **FormCartoGpsx\_FormClosing** : sauvegarde des paramètres.
* Méthodes **Degree**, **SetMap**, **SetGps**, **SetGpx**, **EraseGps**, **SetGpx**, **EraseGpx**, **PortOpen**, **PortClose** et **GetPortName**.
* Méthode **TsslGps\_Click** et **TsslGpxClick** pour afficher ou non la position courante et la trace.
* Surcharge de **WndProc** pour détecter la déconnexion ou la reconnexion du capteur GNSS sur le port USB, comme dans **CartoGPS**, en interceptant les messages **WM\_DEVICECHANGE**[[25]](#footnote-26).

// FormCartoGpsx.cs

**using** System**;**

**using** System**.**Collections**.**Generic**;**

**using** System**.**Drawing**;**

**using** System**.**Windows**.**Forms**;**

**using** System**.**IO**;**

**using** GMap**.**NET**;**

**using** GMap**.**NET**.**WindowsForms**;**

**using** GMap**.**NET**.**WindowsForms**.**Markers**;**

**using** System**.**Data**;**

**using** System**.**Linq**;**

**using** ORMi**;**

**namespace** CartoGpsx

**{**

**[**WMIClass**(**"Win32\_PnPEntity"**)]**

**public** class Device

**{**

// Pour GetPortName()

**public** string Name **{** **get;** **set;** **}**

**}**

**public** **partial** class FormCartoGpsx **:** Form

**{**

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

// Variables globales //\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**private** int numMap**;** // Numéro de carte affichée : 0 => Google sat, 1 => OpenTopoMap

**private** bool okGps**;** // Vrai si position GPS à afficher

**private** bool okGpx**;** // Vrai si route GPS à afficher

**private** String gpxFileName**;** // Nom du fichier Gpx à afficher

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

// Init et End //\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**public** FormCartoGpsx**()**

**{**

InitializeComponent**();**

WindowState **=** FormWindowState**.**Maximized**;**

**}**

**private** void FormCartoGpsx\_Load**(object** sender**,** EventArgs e**)**

**{**

// Port série

**try**

**{**

port**.**PortName **=** GetPortName**();**

**}**

**catch** **{** **}**

tsslIco**.**Image **=** Resource**.**ko**;**

PortOpen**();**

// Carte

numMap **=** Set**.**Default**.**numMap**;**

SetMap**(**numMap**);**

GMaps**.**Instance**.**Mode **=** AccessMode**.**ServerAndCache**;**

gmap**.**CacheLocation **=** @".\cache\"**;**

gmap**.**MapScaleInfoEnabled **=** **true;**

gmap**.**Zoom **=** Set**.**Default**.**zoomMap**;**

gmap**.**Size **=** **new** Size**(this.**Width**,** **this.**Height**);**

gmap**.**Position **=** **new** PointLatLng**(**Set**.**Default**.**latCenterMap**,** Set**.**Default**.**lngCenterMap**);**

gmap**.**ShowCenter **=** **false;**

// Overlay avec marqueur et route

GMapOverlay ovl **=** **new** GMapOverlay**(**"ovl"**);**

gmap**.**Overlays**.**Add**(**ovl**);**

GMapMarker marker **=** **new** GMarkerGoogle**(new** PointLatLng**(),** GMarkerGoogleType**.**red\_dot**);**

marker**.**IsVisible **=** **false;**

ovl**.**Markers**.**Add**(**marker**);**

GMapRoute route **=** **new** GMapRoute**(new** List**<**PointLatLng**>(),** ""**);**

route**.**Stroke **=** **new** Pen**(**Color**.**Red**,** 4**);**

route**.**IsVisible **=** **false;**

ovl**.**Routes**.**Add**(**route**);**

okGps **=** Set**.**Default**.**okGps**;**

okGpx **=** Set**.**Default**.**okGpx**;**

**if** **(!**okGps**)** EraseGps**();**

**if** **(**okGpx**)**

**{**

gpxFileName **=** Set**.**Default**.**gpxFileName**;**

SetGpx**(**gpxFileName**);**

**}**

**}**

**private** void FormCartoGpsx\_FormClosing**(object** sender**,** FormClosingEventArgs e**)**

**{**

Set**.**Default**.**numMap **=** numMap**;**

Set**.**Default**.**zoomMap **=** gmap**.**Zoom**;**

Set**.**Default**.**latCenterMap **=** gmap**.**Position**.**Lat**;**

Set**.**Default**.**lngCenterMap **=** gmap**.**Position**.**Lng**;**

Set**.**Default**.**okGps **=** okGps**;**

Set**.**Default**.**okGpx **=** okGpx**;**

Set**.**Default**.**gpxFileName **=** gpxFileName**;**

Set**.**Default**.**Save**();**

**}**

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

// Map //\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

void SetMap**(**int numMap**)**

**{**

**switch** **(**numMap**)**

**{**

**case** 0**:**

gmap**.**MapProvider **=** GMap**.**NET**.**MapProviders**.**GoogleSatelliteMapProvider**.**Instance**;**

Text **=** "CartoGPSX : Google Satellite"**;**

**break;**

**case** 1**:**

gmap**.**MapProvider **=** GMap**.**NET**.**MapProviders**.**OpenStreetMapProvider**.**Instance**;**

Text **=** "CartoGPSX : Open Topo Map"**;**

**break;**

**}**

**}**

**private** void BtnChangeMap\_Click**(object** sender**,** EventArgs e**)**

**{**

numMap **=** 1 **-** numMap**;**

SetMap**(**numMap**);**

**}**

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

// Gps //\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

double Degree**(**string str**)**

**{**

double a **=** double**.**Parse**(**str**.**Replace**(**'.'**,** ','**))** **/** 100**;**

double e **=** Math**.**Truncate**(**a**);**

**return** e **+** **(**a **-** e**)** **/** 0.6**;**

**}**

void SetGps**(**double lon**,** double lat**,** float alt**,** int sat**,** float hdop**)**

**{**

tsslLon**.**Text **=** "Lon : " **+** lon**.**ToString**(**"#.####0°"**);**

tsslLat**.**Text **=** "Lat : " **+** lat**.**ToString**(**"#.####0°"**);**

tsslAlt**.**Text **=** "Alt : " **+** alt**.**ToString**(**"###m"**);**

tsslSat**.**Text **=** "Sat : " **+** sat**.**ToString**(**"#"**);**

tsslHDOP**.**Text **=** "HDOP : " **+** hdop**.**ToString**(**"0.#0"**);**

gmap**.**Overlays**[**0**].**Markers**[**0**].**Position **=** **new** PointLatLng**(**lat**,** lon**);**

gmap**.**Overlays**[**0**].**Markers**[**0**].**IsVisible **=** **true;**

**}**

void EraseGps**()**

**{**

**try** **{** gmap**.**Overlays**[**0**].**Markers**[**0**].**IsVisible **=** **false;** **}**

**catch** **{** **}**

tsslLon**.**Text **=** ""**;**

tsslLat**.**Text **=** ""**;**

tsslAlt**.**Text **=** ""**;**

tsslSat**.**Text **=** ""**;**

tsslHDOP**.**Text **=** ""**;**

**}**

**private** void TsslGps\_Click**(object** sender**,** EventArgs e**)**

**{**

**if** **(!**port**.**IsOpen**)** **return;**

okGps **=** **!**okGps**;**

**if** **(!**okGps**)** EraseGps**();**

**}**

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

// Gpx //\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**private** void SetGpx**(**String strFileName**)**

**{**

**try**

**{**

gpxType gpx **=** gmap**.**Manager**.**DeserializeGPX**(**File**.**ReadAllText**(**strFileName**));**

**if** **(**gpx **!=** **null)**

**{**

**if** **(**gpx**.**trk**.**Length **>** 0**)**

**{**

gmap**.**Overlays**[**0**].**Routes**[**0**].**Points**.**Clear**();**

**foreach** **(**var trk **in** gpx**.**trk**)**

**{**

List**<**PointLatLng**>** points **=** **new** List**<**PointLatLng**>();**

**foreach** **(**var seg **in** trk**.**trkseg**)**

**{**

**foreach** **(**var p **in** seg**.**trkpt**)**

**{**

points**.**Add**(new** PointLatLng**((**double**)**p**.**lat**,** **(**double**)**p**.**lon**));**

**}**

**}**

gmap**.**Overlays**[**0**].**Routes**[**0**].**Points**.**AddRange**(**points**);**

**}**

tsslGpx**.**Text **=** Path**.**GetFileName**(**strFileName**);**

gmap**.**Overlays**[**0**].**Routes**[**0**].**IsVisible **=** **true;**

// gmap.ZoomAndCenterRoutes(null);

okGpx **=** **true;**

gpxFileName **=** strFileName**;**

**}**

**}**

**}**

**catch**

**{**

**}**

**}**

**private** void TsslGpx\_Click**(object** sender**,** EventArgs e**)**

**{**

okGpx **=** **!**okGpx**;**

**switch** **(**okGpx**)**

**{**

**case** **true:**

**using** **(**FileDialog dlg **=** **new** OpenFileDialog**())**

**{**

dlg**.**CheckPathExists **=** **true;**

dlg**.**CheckFileExists **=** **false;**

dlg**.**AddExtension **=** **true;**

dlg**.**DefaultExt **=** "gpx"**;**

dlg**.**ValidateNames **=** **true;**

dlg**.**Title **=** "Afficher une route .Gpx"**;**

dlg**.**Filter **=** "gpx files (\*.gpx)|\*.gpx"**;**

dlg**.**FilterIndex **=** 1**;**

dlg**.**RestoreDirectory **=** **true;**

**if** **(**dlg**.**ShowDialog**()** **==** DialogResult**.**OK**)** SetGpx**(**dlg**.**FileName**);**

**}**

**break;**

**case** **false:**

gmap**.**Overlays**[**0**].**Routes**[**0**].**IsVisible **=** **false;**

tsslGpx**.**Text **=** ""**;**

**break;**

**}**

**}**

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

// Port série //\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**private** String GetPortName**()**

**{**

String str **=** ""**;**

WMIHelper helper **=** **new** WMIHelper**(**"root\\CimV2"**);**

Device device **=** helper**.**Query**<**Device**>().**ToList**()**

**.**Where**(**p **=>** **(**p**.**Name **??** ""**)**

**.**Contains**(**"u-blox GNSS Receiver"**)).**SingleOrDefault**();**

**if** **(**device **!=** **null)**

**{**

str **=** device**.**Name**;**

int i **=** str**.**IndexOf**(**"("**);**

int j **=** str**.**LastIndexOf**(**")"**);**

str **=** str**.**Substring**(**i **+** 1**,** j **-** i **-** 1**);**

**}**

**return** str**;**

**}**

**private** void PortOpen**()**

**{**

**try**

**{**

port**.**Open**();**

tsslIco**.**Image **=** Resource**.**ok**;**

okGps **=** **true;**

**}**

**catch**

**{**

EraseGps**();**

**}**

**}**

**private** void PortClose**()**

**{**

**try**

**{**

**if** **(**port**.**IsOpen**)** port**.**Close**();**

tsslIco**.**Image **=** Resource**.**ko**;**

EraseGps**();**

**}**

**catch**

**{**

**}**

**}**

**protected** **override** void WndProc**(ref** Message m**)**

**{**

const int WM\_DEVICECHANGE **=** 0x0219**;**

const int DBT\_DEVICEARRIVAL **=** 0x8000**;**

const int DBT\_DEVICEREMOVECOMPLETE **=** 0x8004**;**

**if** **(**m**.**Msg **==** WM\_DEVICECHANGE**)**

**{**

**switch** **(**m**.**WParam**.**ToInt32**())**

**{**

**case** DBT\_DEVICEARRIVAL**:**

PortOpen**();**

**break;**

**case** DBT\_DEVICEREMOVECOMPLETE**:**

PortClose**();**

**break;**

**}**

**}**

**base.**WndProc**(ref** m**);**

**}**

**private** void Port\_DataReceived**(object** sender**,** System**.**IO**.**Ports**.**SerialDataReceivedEventArgs e**)**

**{**

**try**

**{**

**if** **(**port**.**BytesToRead **<** 90**)** **return;**

string str **=** port**.**ReadExisting**();**

**if** **(!**okGps**)** **return;**

int deb **=** str**.**IndexOf**(**"$GNGGA"**);**

**if** **(**deb **>=** 0**)**

**{**

int fin **=** str**.**IndexOf**(**'\n'**,** deb**);**

**if** **(**fin **>** deb**)**

**{**

str **=** str**.**Substring**(**deb**,** fin **-** deb **+** 1**);**

string**[]** tabStr **=** str**.**Split**(**','**);**

double lon **=** Degree**(**tabStr**[**4**]);**

double lat **=** Degree**(**tabStr**[**2**]);**

float alt **=** float**.**Parse**(**tabStr**[**9**].**Replace**(**'.'**,** ','**));**

int sat **=** int**.**Parse**(**tabStr**[**7**]);**

float hdop **=** float**.**Parse**(**tabStr**[**8**].**Replace**(**'.'**,** ','**));**

**if** **(**okGps**)** SetGps**(**lon**,** lat**,** alt**,** sat**,** hdop**);**

**}**

**}**

**}**

**catch**

**{**

**}**

**}**

**}**

**}**

Nous avons également modifié très légèrement le code de la bibliothèque **GMap** de manière à :

* Afficher les tuiles **Open Topo Map**.
* Afficher plus clairement l’échelle.

**GMap.NET.Core.GMap.NET.MapProviders.OpenStreetMap.OpenStreetMapProvider.cs :**

595 static **readonly** string UrlFormat **=** "https://{0}.tile.open**topo**map.org/{1}/{2}/{3}.png"**;**

**GMap.NET.WindowsForms.GMapControl.cs :**

192 **public** Pen ScalePen **=** **new** Pen**(**Brushes**.**Red**,** 3**);**

539 Font ScaleFont **=** **new** Font**(**FontFamily**.**GenericSansSerif**,** 10**,** FontStyle**.**Bold|FontStyle**.**Italic**);**

1541 **if** **(**MapScaleInfoEnabled**)**

**{**

**if** **(**Width **>** Core**.**pxRes5000km**)**

**{**

g**.**DrawRectangle**(**ScalePen**,** 10**,** 10**,** Core**.**pxRes5000km**,** 20**);**

g**.**DrawString**(**"5000Km"**,** ScaleFont**,** Brushes**.Red,** Core**.**pxRes5000km **+** 10**,** 11**);**

**}**

**if** **(**Width **>** Core**.**pxRes1000km**)**

**{**

g**.**DrawRectangle**(**ScalePen**,** 10**,** 10**,** Core**.**pxRes1000km**,** 20**);**

g**.**DrawString**(**"1000Km"**,** ScaleFont**,** Brushes**.Red,** Core**.**pxRes1000km **+** 10**,** 11**);**

**}**

**if** **(**Width **>** Core**.**pxRes100km **&&** Zoom **>** 2**)**

**{**

g**.**DrawRectangle**(**ScalePen**,** 10**,** 10**,** Core**.**pxRes100km**,** 20**);**

g**.**DrawString**(**"100Km"**,** ScaleFont**,** Brushes**.Red,** Core**.**pxRes100km **+** 10**,** 11**);**

**}**

**if** **(**Width **>** Core**.**pxRes10km **&&** Zoom **>** 5**)**

**{**

g**.**DrawRectangle**(**ScalePen**,** 10**,** 10**,** Core**.**pxRes10km**,** 20**);**

g**.**DrawString**(**"10Km"**,** ScaleFont**,** Brushes**.Red,** Core**.**pxRes10km **+** 10**,** 11**);**

**}**

**if** **(**Width **>** Core**.**pxRes1000m **&&** Zoom **>=** 10**)**

**{**

g**.**DrawRectangle**(**ScalePen**,** 10**,** 10**,** Core**.**pxRes1000m**,** 20**);**

g**.**DrawString**(**"1000m"**,** ScaleFont**,** Brushes**.Red,** Core**.**pxRes1000m **+** 10**,** 11**);**

**}**

**if** **(**Width **>** Core**.**pxRes100m **&&** Zoom **>** 11**)**

**{**

g**.**DrawRectangle**(**ScalePen**,** 10**,** 10**,** Core**.**pxRes100m**,** 20**);**

g**.**DrawString**(**"100m"**,** ScaleFont**,** Brushes**.Red,** Core**.**pxRes100m **+** 9**,** 11**);**

**}**

1573 **}**

## Utilisation avec une tablette sous Windows 10[[26]](#footnote-27)





## Conclusion et perspectives

La bibliothèque **GMap** a ici permis de simplifier considérablement la programmation et l’utilisation de cette application, et tout particulièrement en ce qui concerne l’accès aux tuiles cartographiques de différents types de cartes, leur mise en cache automatique qui les rend disponibles hors connexion et leur affichage à l’écran. De plus, la technique des overlays est très simple à mettre en œuvre pour afficher des marqueurs et des traces GPX en superposition.

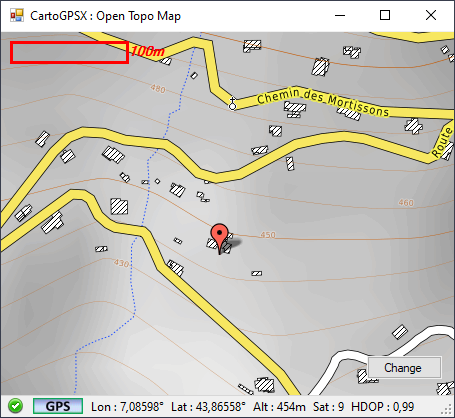
Notre objectif principal étant d’implémenter ces fonctionnalités de la manière la plus directe possible, nous n’avons pas cherché à exploiter toutes les autres possibilités de cette bibliothèque telles que :

* Stockage d’une portion de carte.
* Enregistrement automatique d’une trace.
* Marqueur circulaire pour indiquer la position géographique, comme dans CartoGPS.

Et il aurait été plus pertinent d’ajouter un nouveau provider pour accéder aux cartes **Open Topo Map**, plutôt que de modifier le code du provider **Open Street Map**.

A noter aussi un petit bug au niveau du zoom, du fait que sa valeur maximum pour les tuiles **Open Topo Map** (17) est plus faible que pour les tuiles **Google Satellite** (20). Ceci ne se pose qu’au niveau du chargement car après, un zoom « bitmap » est appliqué. Il suffit donc de réduire le facteur de zoom pour permettre le chargement, quitte à l’augmenter ensuite :

Enfin, au niveau GNSS, on pourrait mettre à profit l’usage de son mode « économie d’énergie », avec une commande **UBX-CFG-PM2**[[27]](#footnote-28), pour réduire la consommation de la batterie de la tablette. Mais comme on peut aussi la mettre en veille automatiquement au bout d’un certain temps d’inactivité, coupant ainsi l’alimentation du GNSS, ce n’est pas indispensable.

1. GNSS = Géolocalisation et Navigation par un Système de Satellites. Souvent appelé à tort GPS, car ceci désigne un des systèmes satellitaires possibles. [↑](#footnote-ref-2)
2. J. Lemaire : CartoGPS (2016). [↑](#footnote-ref-3)
3. <https://www.geoportail.gouv.fr/> [↑](#footnote-ref-4)
4. <http://alpinequest.net/fr> [↑](#footnote-ref-5)
5. <https://wiki.openstreetmap.org/wiki/FR:OpenTopoMap> [↑](#footnote-ref-6)
6. Mois de 8€/an ! [↑](#footnote-ref-7)
7. <https://fr.wikipedia.org/wiki/GPX_(format_de_fichier)> [↑](#footnote-ref-8)
8. <https://www.codeproject.com/Articles/32643/GMap-NET-Great-Maps-for-Windows-Forms-and-Presenta> [↑](#footnote-ref-9)
9. <https://github.com/radioman/greatmaps> [↑](#footnote-ref-10)
10. <http://www.independent-software.com/category/gmap.html> [↑](#footnote-ref-11)
11. <https://www.youtube.com/playlist?list=PLlD7n_T-mUjVuqIhWVfaNhnpqCZmNcA9e> [↑](#footnote-ref-12)
12. <https://github.com/radioman/greatmaps> [↑](#footnote-ref-13)
13. Clic-droit pour déplacer, molette pour zoomer. [↑](#footnote-ref-14)
14. <https://store.drotek.com/DP0107> [↑](#footnote-ref-15)
15. <https://www.u-blox.com/sites/default/files/NEO-M8-FW3_DataSheet_%28UBX-15031086%29.pdf> [↑](#footnote-ref-16)
16. Par exemple avec MyGPSFiles (<https://www.mygpsfiles.com/app/>). [↑](#footnote-ref-17)
17. Par exemple avec l’application Android **AlpineQuest** (<http://alpinequest.net/fr>) sur un smartphone disposant d’un capteur GNSS intégré. [↑](#footnote-ref-18)
18. Par exemple avec GPX2KML (<https://gpx2kml.com/>). [↑](#footnote-ref-19)
19. <https://www.u-blox.com/en/product/u-center> [↑](#footnote-ref-20)
20. <https://www.u-blox.com/en/docs/UBX-13003221> [↑](#footnote-ref-21)
21. <https://github.com/nicoriff/ORMi> [↑](#footnote-ref-22)
22. Windows Management Instrumentation. [↑](#footnote-ref-23)
23. On peut également utiliser la class **System.Management**. [↑](#footnote-ref-24)
24. <https://github.com/nicoriff/ORMi> [↑](#footnote-ref-25)
25. <https://docs.microsoft.com/fr-fr/windows/win32/devio/wm-devicechange?redirectedfrom=MSDN> [↑](#footnote-ref-26)
26. HP Stream 7 [↑](#footnote-ref-27)
27. <https://www.u-blox.com/en/docs/UBX-13003221> [↑](#footnote-ref-28)