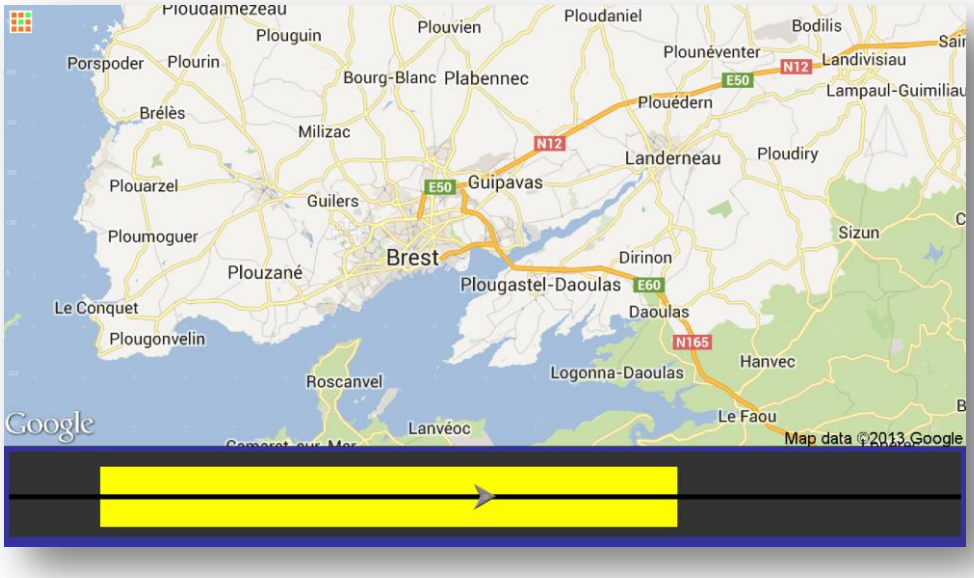
Intégration de la cartographie Google dans le projet *uav3i*

Philippe TANGUY
Document créé le 09 septembre 2013
Dernière modification le 09 septembre 2013



Lancement de l'application

Le lancement de l'application s'effectue à l'aide de la classe *Launcher* (package *com.deev.interaction.uav3i.ui*). Cette classe possède deux fonctions essentielles :

1. lancer l'IHM de l'application (classe *MainFrame*) ;
2. lancer le système de récupération de données vidéos (non encore implémenté).

Note : le lancement de la classe *MainFrame* se fait de manière un peu particulière pour les personnes non habituées à la programmation des interfaces graphiques en Swing. Cette manière de faire permet d'éviter des problèmes d'inter-blocage dans la gestion des événements sur l'IHM de l'application. Plus d'infos sur le tutorial *Threads et performance avec Swing* : http://gfx.developpez.com/tutoriel/java/swing/swing-threading/

L'IHM de l'application

La classe principale est la classe *MainFrame* (package *com.deev.interaction.uav3i.ui*). Elle génère l'interface graphique en instanciant un certain nombre de classes. L'IHM est divisée en deux parties :

- La visualisation cartographique qui possède deux fonctions :
  - Manipulation de la carte (pan + zoom).
  - Définition des centres d'intérêt à prendre en compte dans la mission de surveillance.
- L'IHM de gestion du flux vidéo (*TimeLine*).

L'IHM de gestion des missions et celle du flux vidéo n'est pas couvert dans cette documentation.

Principes d'affichage et gestion de l'interaction avec le fond de carte

La gestion de l'affichage des cartes Google est une adaptation au modèle de programmation du projet *uav3I* d'un précédent projet : *TestGoogleStaticMapAPI*.

Deux composants graphiques sont nécessaires (dans le sens interface graphique *Swing* du terme, ils héritent tous de la classe *JComponent*) pour l'affichage et la gestion de l'interaction utilisateur.

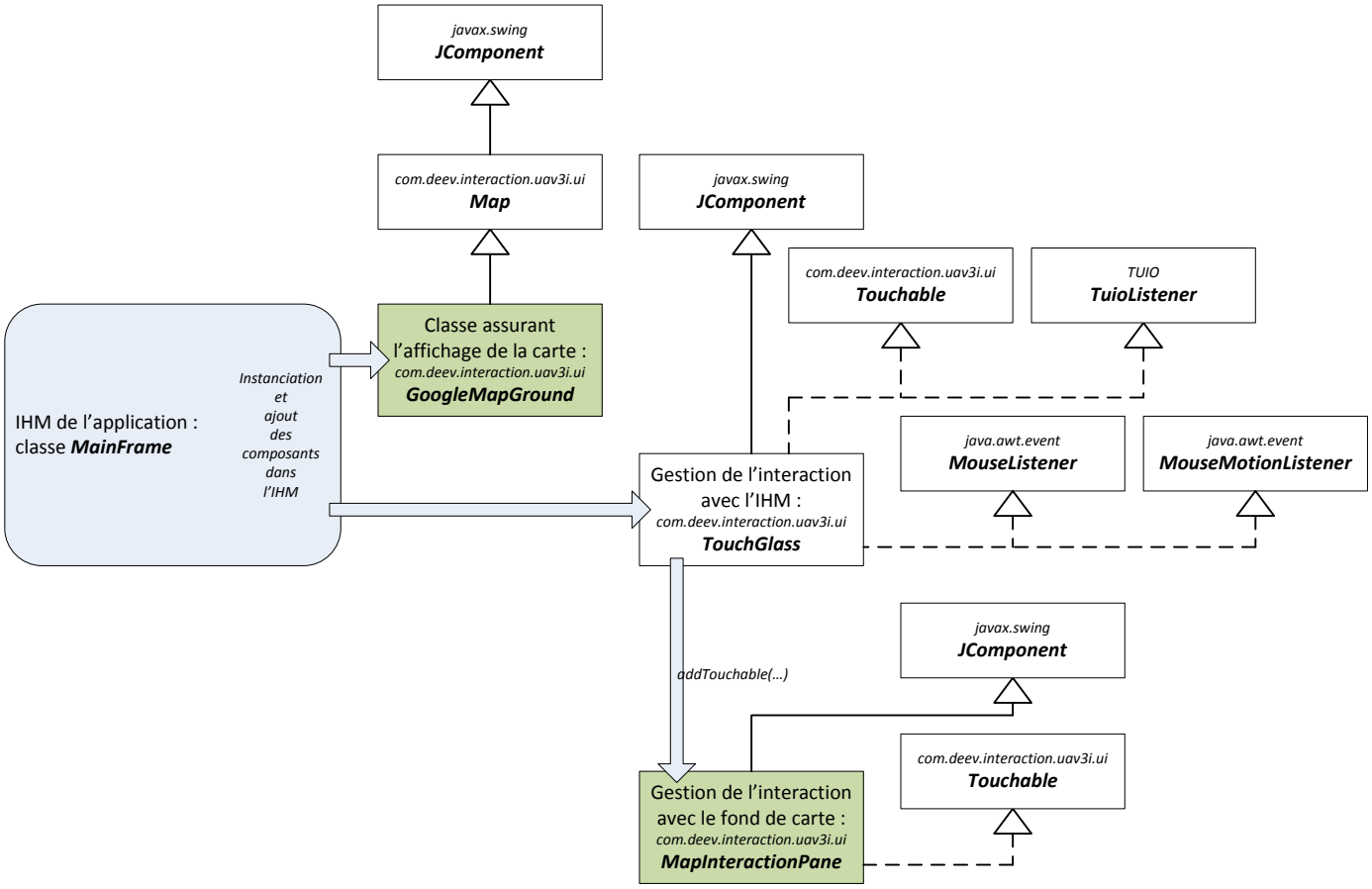


Figure 1 : classes principales pour l'affichage et la gestion de l'interaction.

- Le premier assure l'affichage proprement dit de la carte à l'aide de la classe ***GoogleMapGround*** (package *com.deev.interaction.uav3i.ui*). Son rôle est d'afficher les images au format bitmap de la zone géographique choisie après la récupération auprès des serveurs Google.
  Cette classe hérite de de la classe *Map* (package *com.deev.interaction.uav3i.ui*) qui définit un certain nombre de méthodes qu'il est possible de redéfinir dans la sous-classe.
  La classe *Map* hérite de la classe *JComponent* (package *javax.swing*) : la classe *GoogleMapGround* est donc un composant graphique qu'il est possible d'intégrer dans une IHM swing.

- Le second assure la gestion de l'interaction entre le fond de carte et l'utilisateur, classe ***MapInteractionPane*** (package *com.deev.interaction.uav3i.ui*). À la date de rédaction de cette documentation, la seule fonctionnalité utilisable est encore le déplacement sur le fond de carte.
  Cette classe hérite de *JComponent* (c'est donc un composant graphique) et implémente l'interface *Touchable* (package *com.deev.interaction.uav3i.ui*). Elle doit donc implémenter un certain nombre de méthodes spécifiques à l'interaction utilisateur (*addTouch*, *removeTouch*, etc.).
  D'autres composants graphiques sont aussi susceptibles d'interagir avec l'utilisateur (c'est le cas de la classe *TimeLine*), ils disposent aussi d'un composant spécifique pour la gestion de l'interaction (qui implémente aussi *Touchable*). L'ensemble de ces composants d'interaction est géré par une classe « conteneur » qui est

responsable de la propagation des événements sur l'ensemble de l'IHM au composant spécifique auquel l'événement est dédié (elle les capte en premier lieu puis les distribue au bon composant d'interaction). C'est le rôle de la classe *TouchGlass* (package *com.deev.interaction.uav3i.ui*). Cette classe est instanciée dans la classe *MainFrame*, puis après instanciation des composants d'interaction, ceux-ci sont enregistrés dans la classe TouchGlass par l'appel de la méthode *addTouchable(Touchable t)*.

Une classe d'importance secondaire est aussi instanciée dans la classe *MainFrame*, il s'agit de la classe assurant l'affichage d'un indicateur de téléchargement des carte Google : il s'agit de la classe ***GooggleMapManagerUI*** (package *com.deev.interaction.uav3i.googleMap*).
L'état de téléchargement est indiqué par la présence de 9 carrés (correspondants aux 9 cartes à télécharger) : les cartes déjà chargées sont indiquées à l'aide d'un carré vert et celle encore à télécharger apparaissent en rouge. Quand le téléchargement est terminé, le composant s'estompe (grâce à une magnifique animation !!!) pour disparaître complétement.
Cette classe est instanciée préalablement à l'instanciation des deux classes principales puis passée en paramètre aux constructeurs de celles-ci.

Gestion de l'interaction

Pour le moment, l'interaction sur le fond de carte avec utilisateur n'a été testée qu'avec la souris. Cette interaction est d'abord gérée dans la classe *TouchGlass* qui, outre l'interface *Touchable*, implémente aussi les interfaces *MouseListener* et *MouseMotionListener*. Les événements souris sont ensuite traduits en événements tactiles par l'appel des méthodes de la classe *Touchable* sélectionnée : *mousePressed* appelle *addTouch*, *mouseDragged* appelle *updateTouch*, etc.
Le fonctionnement avec un écran tactile (non encore testé) devrait être similaire, les méthodes de l'interface *TuioListener* (*addTuioCursor*, *removeTuioCursor*, etc.) appellent aussi les méthodes équivalentes de l'interface *Touchable*.
➔à tester

La responsabilité de l'interaction appartient à la classe *MapInteractionPane*. Cette classe est de type *Touchable*, les méthodes suivantes doivent obligatoirement être définies :

- *float* ***getInterestForPoint****(float x, float y)*
  Cette méthode permet à un composant d'interaction d'exprimer sa priorité lors de l'interaction d'un utilisateur sur un point de coordonnées (x, y).
  Dans le cas de la classe *MapInteractionPane*, la valeur renvoyée (100.0) est la valeur la plus haute de l'ensemble des valeurs renvoyées par ce type de composant afin d'être sûr de capter l'événement. De fait, la gestion de zones d'intérêt pour une mission donnée (classe *FingerPane*) est complètement débrayée pour le moment : la classe *FingerPane* renvoie la valeur 10.0 et n'obtient donc jamais la gestion de l'événement.
  ➔à régler

- *void* ***addTouch****(float x, float y, Object touchref)*
  Cette méthode est appelée quand un touché est détecté sur l'IHM. Cette méthode n'est pas utilisée dans la classe *MapInteractionPane* : voir méthode *updateTouch*.

- *void* ***updateTouch****(float x, float y, Object touchref)*
  Cette méthode est appelée lorsqu'un déplacement de la carte est voulu (pan). L'ordre des appels successifs est le suivant *TouchGlass.mouseDragged -> TouchGlass.updateTouch -> MapInteractionPane.updateTouch*.
  Deux cas de figure peuvent se présenter :
  - Le déplacement n'a pas encore débuté, un drapeau (attribut *panStarted*) est alors positionné à true. La position de départ (x et y) est alors mémorisée (attributs *panStartX* et *panStartY*).
    Note : le positionnement à true de ce drapeau engendre le dessin d'un curseur graphique (quadruple

flèche) qui suit le déplacement. L'affichage de ce curseur est réalisé dans la méthode *paintComponent* (voir plus loin).

- Le déplacement a débuté (*panStarted* est déjà positionné à true), on effectue le calcul du décalage en x et en y (mémorisation des valeurs dans les attributs *panDeltaX* et *panDeltaY*) puis on demande au composant d'affichage de mettre à jour la nouvelle position de la carte par l'appel de la méthode *panPx* de la classe *GoogleMapGround*.
  Note : dans ce cas, il n'y a pas de téléchargement de nouvelles cartes, la partie invisible de la carte qui apparait à l'écran est une sous partie des huit cartes additionnelles qui ont été téléchargées autour de la carte principale (nord, nord-est, est, etc.). Le téléchargement n'est déclenché qu'au relâchement du touché (méthode *removeTouch*).

Le corps de la méthode n'est exécuté que si le téléchargement des cartes est terminés, il est bloqué dans le cas contraire et le déplacement est impossible : test conditionnel *if(mapManagerUI.isDrawConpleted())*.

- *void* ***removeTouch****(float x, float y, Object touchref)*
  Cette méthode est appelée à la fin d'un déplacement, elle fera déclencher le téléchargement des cartes à la nouvelle position sur les serveur de Google.
  Elle réalise différentes choses :
  - Repositionnement à false du drapeau *panStarted*.
  - Disparition progressive du curseur de déplacement (quadruple flèche) à l'aide de la classe *ImageLighteningAnim*.
  - Affichage de l'indicateur de téléchargement des cartes (il était invisible depuis le dernier téléchargement).
  - Demande la mise à jour de la carte (demande de téléchargement) par l'appel de la méthode *updateMap* de la classe *GoogleMapGround* avec les dernières valeurs du déplacement mémorisées dans la méthode *updateTouch*.

- *void* ***canceltouch****(Object touchref)*
  Cette méthode n'est pas utilisée dans la classe.

Les autres méthodes :

- Le constructeur de la classe
  Il mémorise dans des attributs privés les références vers les instances des classes créées dans la classe MainFrame : *GoogleMapGround* et *GoogleMapManagerUI*. Il charge aussi dans une instance de *BufferedImage* le curseur de déplacement (quadruple flèche).

- *protected void* ***paintComponent****(Graphics g)*
  Redéfinition de la méthode de la classe *JComponent* : son rôle est uniquement d'afficher les animations nécessaires (estompage de l'indicateur de téléchargement des cartes et du curseur de déplacement) et affichage du curseur.

Gestion de l'affichage de la carte

Note : la version initiale du projet a été livrée avec une classe aux fonctionnalités basiques assurant l'affichage d'un fond de carte au format bitmap sans possibilité de zoom ni de déplacement. Il s'agit de la classe *MapGround* qui affichait un fond de carte de la baie de Douarnenez. Cette classe, toujours présente dans le projet, n'est plus utilisée.

Le composant graphique responsable de l'affichage des cartes est une instance de la classe *GoogleMapGround*. Cette classe assure principalement une fonction d'adaptation entre le modèle de programmation du projet *uav3i* (classe héritant de la classe *Map*, voir figure 1) et les classes précédemment développées dans le projet *TestGoogleStaticMapAPI*. La classe principale, *GoogleMapManager* et les autres classes sont toutes localisées dans le package *com.deev.interaction.uav3i.googleMap*).

Principes de la récupération des cartes sur les serveurs Google

Google rend disponible l'accès à ces cartes de plusieurs manières à l'aide de différentes API. L'utilisation la plus courante est celle utilisant du code *Javascript* au sein de pages Web dynamiques (*Google Maps JavaScript API v3*) qui permet l'accès au contenu géographique. Une autre utilisation, similaire techniquement, est l'intégration de cartes Google au sein d'applications Android ou Iphone.

Aucune API n'existe pour récupérer du contenu géographique directement dans un code Java (ou C, Python, etc.). Une solution, un peu détournée, est d'utiliser l'API Google *Static Maps API V2*. L'objectif premier de cette API est d'offrir la possibilité d'intégrer au sein d'une page Web une image Google Maps obtenue à l'aide d'une URL.

Exemple :

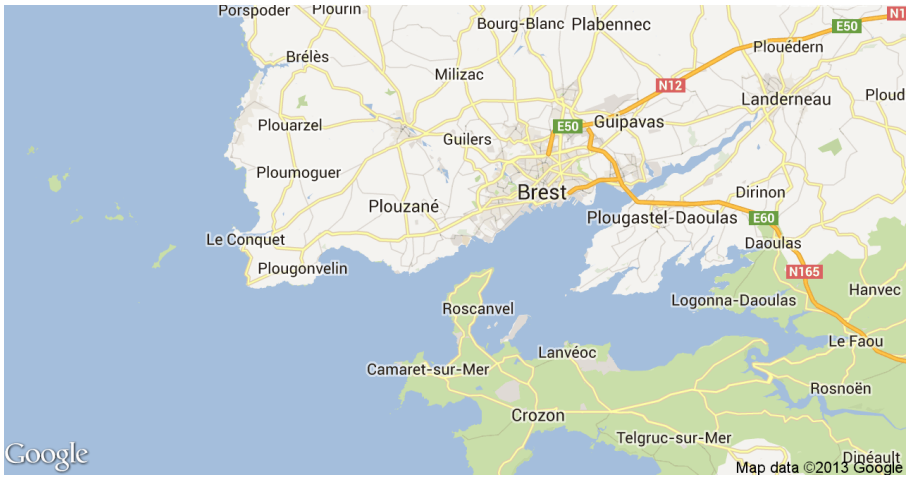


Figure 2 : image obtenue avec l'URL :
http://maps.googleapis.com/maps/api/staticmap?center=48.359407,-4.57013&zoom=10&size=640x333&scale=2&format=png8&maptype=roadmap&sensor=false