Mapzen : Nous utilisons les service de Mapzen (http://mapzen.com), qui proposent une API permettant d'avoir les fichier HGT gratuitement. La classe MapzenManager orchestre le tout et le MapzenGetter s'occupe de gérer un fichier. Ceci nous permet de dessiner toute la terre (sauf les bords que nous avons pas eu le temps de faire marcher). Cela nous permet d'avoir les fichiers HGT dans une zone de rayon 2 autour de l'observateur. Ceci nous permet également a ne plus avoir besoin d'avoir certains fichiers HGT nécessaire, mais au lieu de cela les télécharge. L'inconvénient par contre est qu'une connexion internet est requise. Ce téléchargement a un autre avantage de ne pas avoir a stocker une large quantité de fichier (toute la planète représente presque 2TB de données). Il est conseillé de vider le dossier HGT manuellement de temps à autre, notre programme ne supprimant pas ces fichiers, il peuvent prendre beaucoup de place à la longue.

CustomPainters : Ceci nous as permis de choisir nos peintres ainsi que d'utiliser des peintres de gradients (ou nous définissons une altitude comme une couleur). Nous utilisons plusieurs peintres de gradient inspiré de diverses sources. Nous dessinons le gradient sur un canvas que nous lisons par la suite pour connaitre la couleur à un point souhaité.

Menu Bar : Nous avons rajoute une barre de menu avec un bouton pour enregistrer le panorama, choisir le peintre, choisir un panorama parmi ceux prédéfini, ouvrir la fenêtre pour la vue de dessus. Ainsi qu'un bouton pour quitter.

Trekking profile : Nous avons permis à l'utilisateur d'ouvrir une fenêtre qui présente une vue du dessus de la zone centrée sur la position de l'observateur. Cette vue du dessus est interactive dans la mesure où l'utilisateur peut cliquer sur l'image pour enregistrer une suite de GeoPoint, vider la suite de GeoPoint en utilisant le bouton « Clear », et afficher le profil d'élévation du chemin passant par ces GeoPoint (cf CompositeElevationProfile.java). Cette vue du dessus utilise les interfaces de rendu graphique MapViewRenderer et ElevationProfileRenderer. Elle utilise le peintre noir et blanc, ou le peintre de gradiant si il est sélectionné.

CompositeElevationProfile : Ce patron composite est de type ElevationProfile et définie un profil d'élévation composé de plusieurs profils dont l'azimut n'est pas forcément identique .

ElevationProfile : Cette vue de coté du chemin passant pas les points choisit indique les informations de base, l'effort accompli par une personne qui désirerais le parcourir a pied est calculé en arrière plan mais n'est pas affiché (par question de manque de temps).

Autres fonctionnalisés pas présentes dans le rendu bonus :

Nous avions une classe calculant le chemin optimal entre deux points en tenant compte de l'effort accompli (mis de coté par manque de temps pour l'optimiser, un calcul prend une dizaine voire vingtaine de minutes).

Nous avions une fonctionnalité permettant de dessiner des fonctions mathématiques dans notre programme, ce qui nous a permis de remplacer le Cervin avec un Toblerone. Ainsi qu'une fonction dessinant du "Solid Noise" pour simuler des nuages (https://en.wikipedia.org/wiki/Perlin_noise) Nous n'avions pas le temps d'intégrer ces fonctionnalisées dans le rendu.