Projet : OSM Render

- Rendering of .osm map -

Hucher Ludovic - Lefranc Joaquim - Skoda Jérôme

Les différents fichiers

- ***exemples/ :*** *Quelques cartes au format .osm*
- ***bin/ :*** *Emplacement des exécutables*
- ***resources/ :*** *Emplacement des icons, textures etc*
- ***src/graphic :*** *Sources concernant la partie rendu graphique*
- ***src/model :*** *Sources des types et structures de données*
- ***src/parser :*** *Sources du parser XML*
- ***main.c :*** *Point de démarrage de l'application*
- ***Makefile :*** *Règles de compilation*
- ***styles.txt :*** *Dictionnaire des styles en format texte*

Compilation et exécution

Nettoyage et compilation :

```
$ make clean
$ make
```

Lancement avec une carte en argument :

```
$ make run map=exemples/macarte.osm
```

Librairies nécessaires

- ***xml2 :*** *gestion du parser xml*
- ***SDL2 :*** *graphisme*
- ***SDL2_gfx :*** *fonctions de dessin*
- ***SDL2_ttf :*** *affichage de texte*
- ***SDL2_image :*** *gestion des images*

Fonctionnalités implémentées

- ***Elements de base***
- ***Ordre d'affichage***
- ***Rivières larges***
- ***Feuilles de styles***
- ***Affichage des icones***
- ***Bâtiments creux***
- ***Multipolygons***
- ***Coastlines***
- ***Déplacement et zoom***
- ***Recherche locale***
- ***Export du rendu***
- ***Téléchargement à la volée***
- ***Recherche globale***
- ***Calcul d'itinéraire***
- ***Marquages des noms***

Fonctionnement graphique

Initialisation : *L'initialisation de la SDL se fait dans le* ***main.c*** *après le parsing de la carte. Une nouvelle fenêtre est crée, les paramètres du rendering sont initialisés puis la boucle permettant de capter les événements est lancée.*

Gestion des styles : *Les styles sont gérés grâce à un fichier contenant toutes les informations des styles. La fonction* ***openStyleSheet()*** *lit le fichier passé en argument et construit le dictionnaire de style. Les structures de données associées sont les suivantes.*

Initialisation du dictionnaire :

```
STYLE_ENTRY _dico[DICO_SIZE] = {};
```

Type RGBA_COLOR :

```
typedef struct{
    int r;
    int g;
    int b;
    int a;
} RGBA_COLOR;
```

Type STYLE_ENTRY :

```
typedef struct{
    char *key;
    char *value;
    int weigth;
    RGBA_COLOR color_IN;
    RGBA_COLOR color_OUT;
    char *file_img;
    int priority;
} STYLE_ENTRY;
```

Ordre d'affichage : *Le meilleurs compromis trouvé permettant de gérer la plupart des cas est le suivant. En premier lieu on affiche les membres **outter** des relations. Ensuite vient l'affichage des ways puis les membres **inner** des relations. Les noeuds sont affichés à la fin. L'ordre d'affichage des ways est dicté par l'ordre de lecture des styles dans le fichier. La priorité d'un style est indexée par le numéro de ligne dans le fichier. Le style présent à la première ligne du fichier sera donc affiché en premier.*

Projection et échelle : *La projection est celle de Mercator, grâce aux fonctions disponibles dans la documentation OpenStreetMap. Nous obtenons avec ces formules une conversion des degrés de latitude ou longitude en mètres. Ce qui permet ensuite de calculer l'échelle d'affichage en déterminant le nombre de mètres affichés par un pixel. L'affichage se fait à partir d'un point de référence **(REF_X et REF_Y)** qui est le point milieu de la fenêtre au lancement. Le paramètre **SCALE** lui est déterminé en calculant le ratio X et Y reliant la taille de la fenêtre avec la portion de carte à afficher. Le ratio permettant un recouvrement total de la fenêtre est donc choisi comme échelle d'initialisation.*

Déplacement et zoom : *Le déplacement intervient lorsqu'un événement clavier **(KEY_UP)** est déclenché. Il se fait à l'aide des touches directionnelles. Le zoom lui est sensible aux touches **+** et **-** (pas celles du clavier numérique). Le déplacement et le zoom utilisent des fonctions agissants sur les variables **REF_X, REF_Y** et **SCALE**. La vue est ensuite mise à jour.*

Elements hors cadre : *Les polygons ou tronçons de routes complétement en dehors du cadre de la fenêtre ne sont pas dessinés. Ceci pour eviter les opérations de dessin couteuses.*