Plan article sur TarotClub

0. Intro

- Objectifs du logiciel : modulaire, facilement customisable, portable Linux/Windows/Mac, gratuit et opensource évidemment :)

Technologies utilisées :

Qt 2.3 et plus :
- portabilité du GUI
- Widgets très avancés et certains dédiés au jeu (Qcanvas)
- gratuit
- classes générales très pratiques (Qstring, réseau, XML ...)
- Superbe documentation et large support sur Internet (mailing list etc.)

XML :
- sauvegardes du jeu et des préférences
- format normalisé, clair (fichier texte), et aisément malléable

Lua 5.0 (pour l'IA) :
- gratuit/opensource
- C ANSI -> portabilité assurée
- Machine virtuelle légère
- Très rapide
- facilité d'intégration avec le C/C++
- Langage de script simple

1. Briques de base

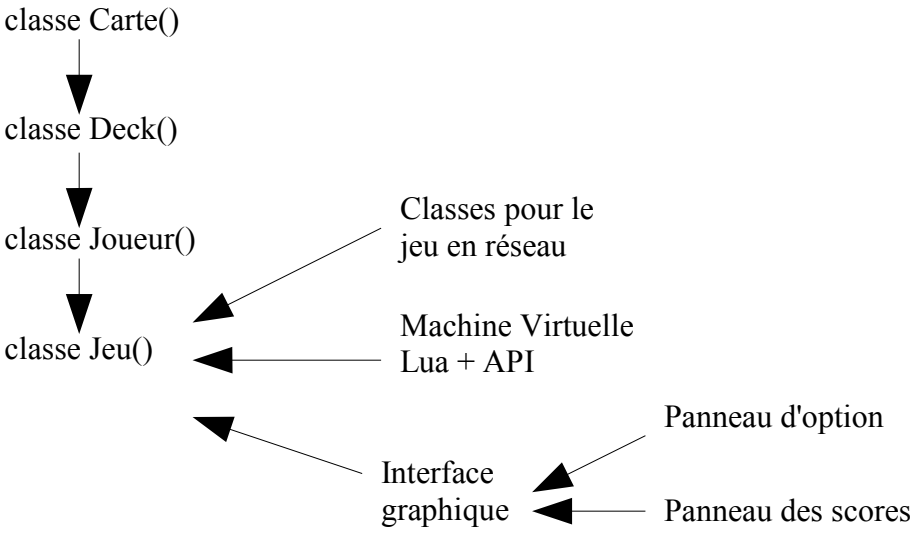
- Objectifs : se familiariser avec Qt et Qcanvas, mettre en place l'interface graphique

- QmainWIndow, Qcanvas au centre
- Découverte de Qsprite, affichage d'une carte à l'écran, différents niveaux du Canvas etc.
- évènements liés à la souris : clics sur le tapis, changement du curseur lors du passage sur la carte, collisions, ressources offertes par Qcanvas pour le jeu.

2. Classes du jeu

Objectifs : organisation des classes, du code source, des objets en mémoire et des données de base

Description du comportement et des données importantes de chaque classe.
Description de l'organisation générale du jeu, des héritages, des classes imbriquées.



Description des données de base :

- Le deck principal
- Le deck du chien
- les variables d'état
- les cartes en mémoire, les listes...

Enumérations C : apportent une abstraction supplémentaire pour manipuler des données (passe, garde etc.

3. Graphiques du jeu

Peu de graphique mais quand même :

- Format PNG partout avec zones transparentes
- Fonctionnalités Qsprite, séquences d'images
- Primitives géométriques, texte
- background

4. Points clés du jeu

- Gestion des séquences du jeu (timers, clics sur le tapis)
- Représentation en mémoire des cartes
  - Qsprite/QList des cartes
  - Id pour chaque carte
  - Utilisation des pointeurs sur des cartes
- Fonctions essentielles du jeu
  - qui a remporté le pli
  - validation d'une carte (respecte-t-elle les règles ?) + modification du curseur en fonction
  - Calcul des points et résultat d'une donne
- Manipulation du XML avec Qt : exemple avec le chargement et la sauvegarde des préférences
- Exemple de customisation de Widget avec Qt : exemple avec QfileDialog + preview de l'image

sélectionnée
- Utiliser QTDesigner : création des fenêtres de score et de préférences avec l'utilitaire graphique, génération de code source à partir des fichiers .ui, particularités entre les plates-formes (taille des fontes variables selon le système)

5. Utilisation des scripts Lua pour l'IA et API

Objectifs : utilité de scripter l'IA, intégrer Lua dans une application, échanger des informations entre un script et l'exécutable hôte.

- Introduction à Lua, historique, utilisation dans des logiciels commerciaux (Baldur's Gate, Homeworld etc.)
- Petit exemple d'intégration de Lua au sein d'un programme C
- Passage de paramètres, caching des erreurs et affichage
- Intégration dans un programme C++ : classe générique, construction d'une classe API, fonctions disponibles à partir du script Lua d'IA.
- Exemple final : création d'une IA simple

6. Code réseau

A faire !

7. Organisation des répertoires/fichiers du projet

8. Conclusion

- Perspectives/améliorations :
  - respect de toutes les règles du jeu de Tarot
  - jeu à 3 et 5 joueurs
  - IA plus évoluée/différents niveaux
  - Portage sur Mac/fichiers de projet

- Customisation
  - cartes
  - tapis de jeu

- Projet sur sourceforge

9. Encarts

- Programme d'instalation NSIS sous Windows
- Génération de la doc HTML à partir de XML + doc de l'API
- NMAKE, options utiles, structure d'un projet Qt, fichier .pro
- Liens : Lua, FFT, TarotClub, Sourceforge, Qt
- Compilation de Lua
- Versions de Qt disponibles sur différents systèmes