Et si je faisais un jeu ?

Ça me démange depuis très très longtemps de refaire un jeu. Mon premier jeu date des années 1984/1985 sur Oric Atmos. J’en avais refait au début des années 2000 sur Windows Mobile. Et depuis : plus rien. C’est décidé, j’en refais un ! Reste à choisir lequel… Que l’on soit clair, je ne me vois pas créer un nouveau concept… je vais refaire un jeu déjà existant. Historiquement, j’ai toujours rêvé devant les screenshots du jeu Cauldron, un jeu des années 80 sortie sur Commodore 64, ZX Spectrum et Amstrad CPC 64 mais jamais sur mon Oric Atmos. En farfouillant le Net, je suis tombé sur cette page qui explique le jeu : <https://www.c64-wiki.com/wiki/Cauldron>

J’ai essayé sur émulateur, Vice et Virtual C64 : le jeu est méga difficile ! Et c’est effectivement les avis que l’on retrouve des différents tests ça et là. Le jeu a même été Tilt d’Or en 185 ! <https://fr.wikipedia.org/wiki/Cauldron>

C’est décidé, ça sera lui ! Mais je me dis que je vais le rendre « plus » jouable et carrément moins difficile, histoire que ça soit fun et non prise de tête.

Les versions sorties sont différentes en fonctions des plateformes, m’a version sera un mélange de ces versions.

Dans la zone aérienne : en scrolling horizontal comme sur Commodore 64.

Dans les grottes : en étant écran par écran, ça oblige les sauts dans l’inconnu, donc il faut apprendre le parcours ou avoir un plan pour jouer. Ça ne me semble pas top. Je vais donc faire en « continue » avec un scrolling dans chaque direction et garder la sorcière au milieu de l’écran, sauf quand on sera sur un bord de la grotte.

Pour le graphisme, je vais oublier la version ZX Spectrum et me baser sur la version C64. Mais comme les couleurs sont un peu fades, je vais surement teinter avec les couleurs de l’Amstrad. Allez c’est parti !

Cauldron : les graphismes

Comme ce qui m’intéresse c’est l’algorithmique et le développement, je vais essayer d’écourter un maximum la phase sur le graphisme.

Deux possibilités s’offrent à moi :

- je fais des copies d’écrans et j’extrais les graphismes un par un.

- j’analyse un peu la structure d’un programme Commodore 64 et j’essai d’extraire les graphismes directement depuis le jeu.

Le C64 utilise de sprites et je suis tombé sur cette page qui explique justement les sprites : <https://www.c64-wiki.com/wiki/Sprite>

Je vais donc tenter ma chance de ce côté et extraire les graphismes directement depuis le jeu.

Je lance un émulateur avec le jeu, tant qu’à faire, j’utilise une version du jeu des hackeurs avec cheat codes intégrés : à moi la vie et l’énergie infinies ! Une fois le jeu démarré, j’utilise la fonctionnalité de sauvegarde d’état de l’émulateur, en espérant que la sauvegarde soit un dump brut de la mémoire. Il faut se rappeler que l’on parle d’un truc genre 64 Ko, donc il n’y a pas vraiment d’intérêt, surtout à notre époque, d’optimiser ça en compressant. Et j’ai effectivement vu juste ! Une entête est ajoutée, mais on retrouve bien le dump mémoire brut de fonderie !

C’est parti, je dégaine mon Visual Studio préféré et crée mon premier projet type « console en C# » pour lire ce dump et tenter d’extraire les graphismes !

Après quelques tâtonnements, c’est un succès !

L’entièreté des graphismes n’est cependant pas sous forme de sprites. Ça s’explique car le C64 a ses limites techniques. Je dois donc également faire des copies d’écrans.

Voici un exemple de graphismes que j’ai compilé sur une planche.

Cauldron – plateforme et moteur graphique

J’ai maintenant le sujet et les premiers graphismes : il est temps de commencer à s’amuser à coder !

Je n’ai pas décidé sur quelle plateforme je veux que ça tourne. Je travaille sur Mac, ça serait plus facile pour moi si ça tournait sur Mac mais je n’ai jamais codé un programme Mac de ma vie ! Par contre je sais faire du multiplateformes avec Xamarin Forms et la version Mac de XF est en béta depuis un moment. Allez, je tente cette piste. C’est parti pour la création de mon premier projet Mac en Xamarin Forms ; la version 3.1 pour être précis.

Maintenant il me faut un « moteur » graphique. Chez Xamarin, ça parle de CocoSharp, de Urho et de SkiaSharp. Les deux premiers me sont vaguement connus et très bien documentés avec une forte communauté derrière eux grâce au nombre d’années d’existences. Mais SkiaSharp me tente bien, c’est assez neuf et me semble très prometteur. Je tente donc cette piste ! Un jour il faudra que je me lance dans Unity 3D… mais avec ce que j’ai a faire, je pense que c’est carrément trop puissant rapport à mon besoin. Jai également hésité avec la lib QT.

Pour me mettre à SkiaSharp, j’ai simplement utilisé cet article de blog : <https://blog.xamarin.com/drawing-with-skiasharp/>

Et c’est un succès, ça fonctionne sur Android, iOS … et sur mon Mac !

Pour information, SkiaSharp est une encapsulation de Skia en dotnet utilisable en C#. Skia est un moteur vectoriel graphique, racheté par Google en 2005, codé en C++ et qui est open source. C’est le moteur graphique de Firefox, Androïd et Chrome. C’est ce qui le rend intéressant à utiliser.

Pour mon développement, la phase d’affichage sera déclenchée par un « InvalidateSurface » et la constitution de l’affichage sera lors de l’événement « PaintSurface ». Cette constitution d’affichage se fera sur un « canvas » sur lequel je peux dessiner du bitmap et effectuer du scaling avec plusieurs algorithmes de resizing. Bref, tout ce dont j’ai besoin.

Structure du programme :

Je charge les divers éléments nécessaires (graphismes par exemple)

Je planifie un calcul du monde du jeu 60 fois par secondes

Je dessine et affiche le monde du jeu dès qu’il y a un changement

Cadeau, un webinar sur Skia et SkiaSharp : <https://channel9.msdn.com/Events/Xamarin/Xamarin-University-Presents-Webinar-Series/SkiaSharp-Graphics-for-XamarinForms>

Cauldron – la carte

A la base, ce jeu est un mélange de deux types de jeu : un shoot-them-all (un shmup) et un jeu de plate-formes. Le shmup est la partie aérienne avec la sorcière qui vole. Le jeu de plate-formes est dans les diverses grottes. Dans les deux cas, je pense gérer ça avec le principe des « tiles ». L’idée est qu’avec un nombre fini de petits graphiques, il est possible de faire des cartes extrêmement grandes. Et pour faire ça, je vais utiliser un outil dont la vie est de gérer ce type de carte et qu’il y a un moment que j’ai envie de découvrir : Tiled Map Editor, maintenant nommé tout simplement mapeditor.

C’est donc parti pour la découverte de ce logiciel très simple mais très puissant sur le sujet. En plus, le format de fichier généré est en json, donc très facilement parsable quelque soit le langage.

Après avoir trouvé une carte complète du jeu fait à partir de copies d’écran de la version ZX Spectrum, j’ai implémenté ma propre version réalisée à partir de copie d’écran de la version C64. Les cartes sont différentes ! Même à ce niveau le jeu n’est pas le même !

Je n’ai pas encore terminé le niveau aérien, mais ça vous donne une bonne idée du concept.

La carte ZX Spectrum : <http://www.gamesdatabase.org/media/sinclair-zx-spectrum/artwork-map/cauldron>

Il ne me restera plus qu’à lire tout ça depuis C# et de l’afficher avec SkiaSharp.

Cauldron – affichage de la carte

On a la les lumières, on a la sono …. Arf non, ça c’est une chanson de Lagaff’… Je repends : j’ai quelques graphismes, j’ai l’outil pour dessiner la carte, il ne me reste plus qu’à mettre tout ça en musique ! Et c’est avec finalement une extrême facilité que j’affiche la carte générée par mapeditor.

La lecture du fichier de mapeditor :

Et voici tout simplement le copier-coller du code réalisant cette opération :

Simple et efficace, la magie opère !

En bonus, je vous mets le code qui gère le scrolling.

1/ la gestion du clavier gauche/droite

2/ le scrolling horizontal en lui-même