Fractal *S*cript

par Adrabi Abderrahim

2010-02-25

0) Introduction

Fractal Script, c'est la partie dynamique d'AiFractals, il se base sur JavaScript-Style ou plutôt QtScript, pour génère des fractales personnalisé par chaque utilisateurs, et ça imagination, il peut support tous les objets de JavaScript comme (Math.random(), etc…) et l'utilisateur il doit obligatoire être un petite peux familiarisé avec la programmation =), et plus tous ça, Fractal Script, il possède un objet spécial pour contrôle les script depuis GUI (interface de l'application), et ce objet se nom « zone » et son but de créer une zone complexe parfait qui offre tous les objets pour création de « son propre script ».

1) Les objets

Les nouveaux objets se sons les objets de l'objet « zone » plutôt pour être honte c'est des méthodes, pour donne le maximum de libre a l'utilisateur, est voici la liste des methodes pour leur paramètres vous consulte la documentation génère par Doxygen la classe « AiZone ».

Méthode	Description
getX	Return le [X] minimale dans la zone complexe enté par l'utilisateur depuis GUI.
getY	Return le [Y] minimale dans la zone complexe enté par l'utilisateur depuis GUI.
getZoneWidth	Return [WIDTH] de la zone complexe enté par l'utilisateur depuis GUI.
getZoneHeight	Return [HEIGHT] de la zone complexe enté par l'utilisateur depuis GUI.
getStartPoint	Return le point de départ pour chaque thread. **ce point est calcule automatique depuis le nombre de threads utilise par utilisateur.**
getIncrement	Return le nombre d'incrémentation pour les points de la zone complexe. **ce nombre de points est calcule automatique depuis le nombre de threads utilise par utilisateur.**
getIteration	Return le nombre d'itération enté par

	l'utilisateur depuis GUI. Il peut être utilise dans déférents endroit dans L'algorithme ca dépend le besoin d'utilisateur.
getPixel	Return Pixel (le color RGB) dans l'image de thread **(n'est pas image globale).** *(n'est pas recommande pour optimisation)*
setPixel	Elle ajoute Pixel soit par ses color (Red-Blue-Green) ou soit par valeur (RGB) dans l'image de thread **(n'est pas image globale).** *(n'est pas recommande pour optimisation)*
setPixels	Il ajoute un tableau de Pixels dans image par ligne, cette methode utilise le point « scanLine ». *(recommande pour optimisation)*
cancel	Cette méthode perme de terminer est sortir des threads et garde les images génère sont qu'il détruise, ca place et dans les boucles elle ce déclenche n'import qu'il moment au l'utilisateur a click sur « le button cancal ». **DANGER : sans cette méthode le programme il ne peut pas sortir jusqu'à la fin.**
progress	Elle permet de change et affiche le progressèrent de programme, c'est l'utilisateur qui définie des value dans script.
getZoom	Utilise pour le zoom, la valeur est calcule automatiquement.
getViewX	Utilise pour le zoom, la valeur est calcule automatiquement.
getViewY	Utilise pour le zoom, la valeur est calcule automatiquement.
Valid	Teste c'est Pixel exists dans l'image de thread. *(n'est pas recommande pour optimisation)*
sendSnapshot	Méthode magique permet de faire des mises à jour de view encours de génération de l'image. « Pour l'animation de view ».
beep	Il envoi un signal sonore. Utilise pour les alertes.
initRandom	Méthode pour initialise le générateur des nombre aléatoire, utilise seulement si utilisateur il ne veut pas utilise random de JavaScript.

generateUInt64	Return random nombre.
generateDouble	Return rendom nombre.
generateUInt32	Return rendom nombre.
getLastX	Return X sauvegardé avec la méthode « setLastXYI ». Utilisation unique pour God-Mode.
getLastY	
getLastIteration	Return Iteration sauvegardé avec la méthode « setLastXYI ». Utilisation unique pour God-Mode.
setLastXYI	Sauvegarde les dernier nombre pour « getLast[XXX] »

2) Exemple Mandelbrot

Cet exemple est une démonstration de création d'un script personnalise, avec l'utilisation de God Mode (voir script « mandelbrot-v2.js ») pour size 400x400:

Pour commence un script il faut avoir obtenir les values initiaux de puis GUI et aussi les valeurs calculé automatiquement comme le point de dépare, les points d'incrémentation:

```
var zheight = zone.getZoneHeight();
var zwidth  = zone.getZoneWidth();
var iheight = zone.getImageHeight();
var iwidth  = zone.getImageWidth();
var minx    = zone.getX();
var miny    = zone.getY();
var startp  = zone.getStartPoint();
var inc     = zone.getIncrement();
var zoom    = zone.getZoom();
var viewx   = zone.getViewX();
var viewy   = zone.getViewY();

var lastY   = zone.getLastY();
```

On ajoute « zone.cancel() » pour chaque long boucle de programme qui génère fractale si on veut que le programme s'arrête au moment de click sur le bouton « arrête » et nous retourne le résultats actuel, notez bien que « God-Mode » lui aussi a besoin de cette méthode pour qu'il contrôle threads, comme :

```
while( y < iheight && !zone.cancel() )
{
        // …
      while( x < iwidth && !zone.cancel() )
      {
                // …
      }
        // …
}
```

Si vous voyez script vous pouvez remarque que j'ai utilise « setPixel », je l'utilise seulement pour la démonstration.

Le code suivant il permet de change la value de progresse bar et faire une mise à jour pour view chaque y mod 4 = 0, euh c'est la magie.

```
if( y % 4 == 0 )
{
        zone.progress( y / 4 );
        zone.sendSnapshot();
}
```

Pour l'utilisation de « God Mode » soit possible il faut sauvegarder les derniers [x] et [y], pour ne pas répété les itérations de le début, le but de « God Mode » c'est sérialisé les threads dans sont état, et la re-utilisation « d'eux » par d'autre utilisateurs, ou un autre moment, et aussi pour l'utilisation de Network rendiring etc... et pour tous ça soit possible il faut seulement ajout une simple méthode « setLastXYI » a la fin de script. Qui donnes ces images :