Théorie des langages et compilation

Projet : QtDrawing

À rendre avant le 01/04/2016 23h59

Sujet

Les parseurs (analayseurs syntaxiques) peuvent être utilisés pour transformer un langage en une structure de données ou encore un autre langage. L'avantage de Lex et Yacc combinés est la simplicité à effectuer ce traitement. Le but de ce projet est de proposer un langage accessible à un non-informaticien permettant de dessiner avec toute la puissance de la bibliothèque Qt tout en donnant la possibilité de créer de petits algorithmes. Pour cela, nous utilisons en entrée un fichier texte avec un langage proche de CSS que nous traitons pour obtenir en sortie une fenêtre dans laquelle nous avons dessiné avec Qt les instructions de l'utilisateur. Dans ce projet nous implémentons seulement une base permettant de dessiner des lignes, des rectangles et des cercles (et pour les plus avancés des images) ainsi que des traitements simples sur ces formes comme la rotation, le changement de couleur, le réglage de l'opacité.

Langage de dessin

Qt étant une bibliothèque d'interface graphique demandant des compétences en programmation, nous souhaitons proposer un langage de dessin programmable simple utilisant cette bibliothèque. Ce langage reprend donc une syntaxe proche de CSS, langage connu pour sa simplicité. Nous proposons dans ce projet seulement 3 formes (la ligne, le rectangle et le cercle) avec leurs caractéristiques (coordonnées, longueur, largeur, rayon) et leurs options (couleur, épaisseur, remplissage, rotation, opacité). Chaque forme pourra être représentée par une variable afin de l'identifier. Voici un premier exemple du langage accepté :

$Rectangle(10, 15, 20, 25)\{rouge, 5, plein, 90°, 50\%\}$

$CERCLE(1, 1, 50, 50)\{$
$couleur = vert$
$rotation = 50°$
$remplissage = vide$
$épaisseur = 3$
$opacité = 90\%$
$\}$

$LigNe\ l1(1, 1, 50, 50)$
$l1[épaisseur] = 4$
$l1[couleur] = bleu$

1 Spécifications

Le programme demandé doit répondre aux spécifications suivantes. Vous devez développer votre programme avec flex et bison en C ou C++ à l'aide de la bibliothèque Qt en s'aidant des sources présentes dans l'archive. Vous pouvez compléter et/ou modifier celles-ci pour obtenir un parser en flex/bison permettant de dessiner avec le nouveau langage dont les spécificités de base demandées (8 points) sont :

— Une forme peut être les chaines de caractères : *rectangle*, *cercle* ou *ligne* (les formes ne sont pas sensibles à la casse donc CeRClE = cercle)

- Les carctéristiques suivent une forme entre parenthèses et sont des entiers.
  - $Ligne(X1, Y1, X2, Y2)$ avec $X1, Y1$ et $X2, Y2$ des coordonées,
  - $Rectangle(X, Y, longueur, hauteur)$ X, Y les coordonnées d'origine et $longueur, hauteur$ respectivement la longueur et la hauteur du rectangle,
  - $Cercle(X, Y, rayon)$ X, Y les coordonées du centre du cercle et $rayon$ son rayon.
- Les options dans l'ordre entre accolades sur une ligne :
  - *couleur* : parmi $rouge, vert, bleu, jaune, noir, blanc, gris$
  - *épaisseur* : un entier définissant l'épaisseur du trait
  - *remplissage* : *plein* ou *vide*
  - *rotation* : rotation de la forme en degré (ex :90°)
  - *opacité* : en pourcentage définissant l'opacité de la forme
- Des commentaires sous deux formes :
  - commentaire sur une ligne \\
  - commentaire sur plusieurs lignes $\backslash * \ mon\ commentaire \ * \backslash$

Spécificités avancées (5 points) :

- Les options peuvent apparaître dans le désordre entre accolades une par ligne sous la forme : $option = valeur$
- Une variable peut être associée à une forme suivi de ses caractéristiques (ex : $LigNe\ l1(1, 1, 50, 50)$) une variable étant une série de lettres et de chiffres commençant par une lettre minuscule.
- Les options entre crochets peuvent être affectées à une variable si elle a été précédemment définie (ex : $l1[couleur] = bleu$)
- Les propriétés de la fenêtre peuvent aussi être modifiées de cette manière (ex :$Fentre[largeur] = 1920$) on pourra modifier aussi la couleur

Encore plus loin ! (5 points)

- Gestion des erreurs de parsing
- Les couleurs en RGB dans le format $\#R, G, B$ avec R,G et B des entiers entre 0 et 255 (ex : $l1[couleur] = \#255, 0, 0$)
- Dessiner une image ! (à vous de décider pour le format en prenant compte qu'il nous faut au moins le chemin de l'image et les coordonnées du point le plus en haut à gauche, pensez à aller voir ce qu'il faut pour utiliser drawImage dans Qt)
- Autoriser l'affectation d'une option d'une variable à une autre (ex :$l1[couleur] = l2[couleur]$)

Bonus (2 points) : proposer la possibilité d'effectuer une boucle pour par exemple tracer 9 carrés sous la forme d'un carré.

Il y aura 2 points sur la propreté et la clareté de l'implémentation. Le code DOIT compiler à l'aide d'un simple script (soit celui présent dans l'archive ou un autre) et le code commenté avec quelques explications. Aucun autre document n'est demandé donc pensez à bien expliquer ce que vous faites en commentaire dans votre code. Le barême est indicatif et est susceptible d'être modifié. Vous enverez une archive de votre projet à l'adresse suivante **fgarreau@info.univ-angers.fr** dont j'accuserai réception, sans réponse de ma part au bout de 24h pensez à me recontacter.

2 Exemple de programme compatible

$Rectangle(10, 15, 20, 25)\{rouge, 5, plein, 90°, 50\%\}$

$CERCLE(1, 1, 50)\{$
$couleur = vert$
$rotation = 50°$
$remplissage = vide$
$épaisseur = 3$
$opacité = 90\%$
$\}$

$LigNe\ l1(1, 1, 50, 50)$
$l1[épaisseur] = 4$
$l1[couleur] = \#0, 0, 255$

$\backslash*$
ceci est un commentaire et doit etre ignore par le parser
$*\backslash$

$Rectangle\ r1(160, 160, 10, 10)$
$r2[couleur] = l1[couleur]$

$\backslash\backslash$*un autre commentaire*

$Fenetre[longueur] = 1280$
$Fenetre[hauteur] = 1024$
$Fentre[couleur] = blanc$