Fiche de projet - Electronix

Version : 1

Type : Serious Game/Simulateur

Auteur : Alan BRETELLLE et Sophie PELLUET

# SOMMAIRE

* [Résumé](#_Résumé_:)
* [Bibliothèques utilisés](#_Bibliothèques_utilisés_:)
* [Fonctionnalités](#_Fonctionnalités_:)
* [Organigramme simplifié](#_Organigramme_simplifié)

# Résumé :

Electronix est un logiciel codé en C qui permet de simuler des circuits électriques. Il possède 2 modes : didacticiel et libre.

* Le mode didacticiel est composé de 15 niveaux permettant de comprendre les différents concepts de l’électricité.
* Le mode libre permet de concevoir librement des schémas électriques afin de tester des connaissances à plus grande échelle.

# Bibliothèques utilisés :

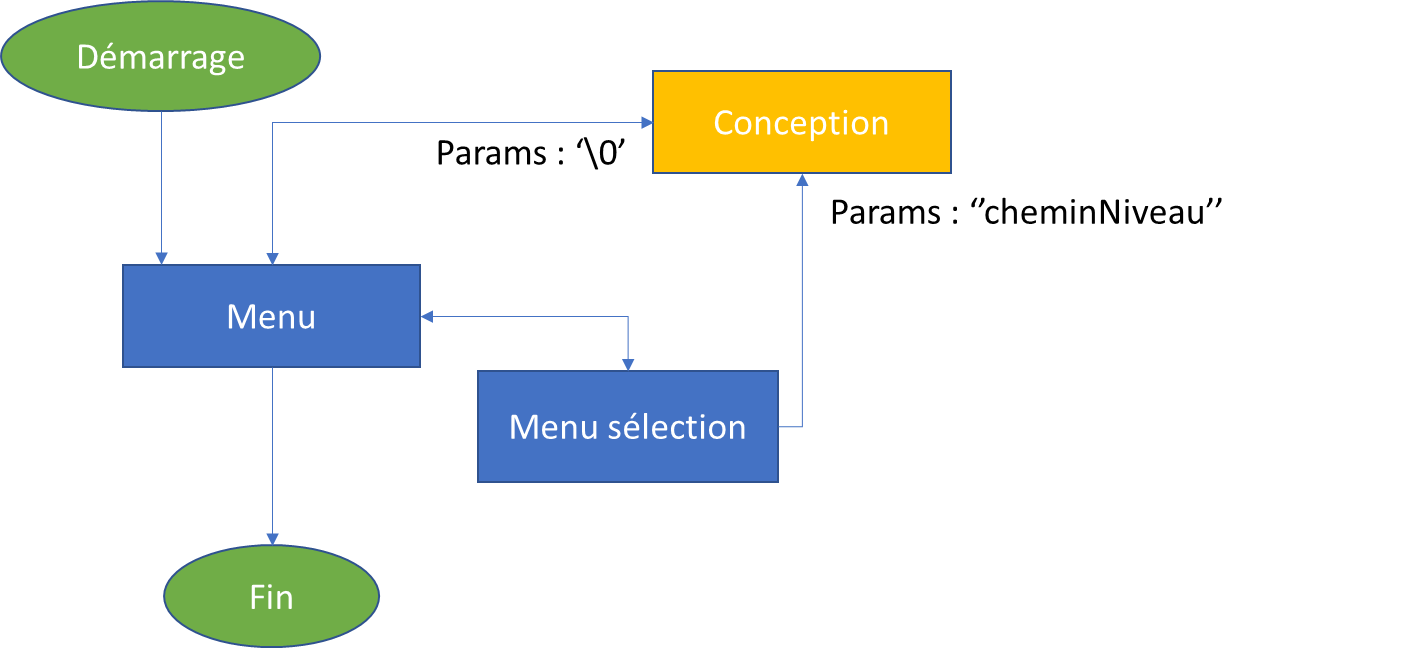
# 

SDL 2 , SDL image, (openGL ?)

# Fonctionnalités :

1. Menu
   1. Principal
      1. Interface : L’utilisateur pourra choisir d’entrer dans 2 modes dès le lancement de l’application. Le titre de l’application, le logo ESGI et le crédit seront affiché en plus des boutons Didacticiel et Libre ainsi qu’un bouton quitter
      2. Evènements : Lors du clique sur le bouton Didacticiel, la page de la sélection du niveau est affichée et sur le bouton, mode Libre, la page conception est affiché avec le paramètre Libre
   2. Sélection du niveau
      1. Initialisation interface : Les signatures des fichiers niveaux seront chargés depuis un fichier. Il contienne des informations utiles sur la nature de chaque niveau : le nom, le chemin, la date de création, la description, la difficulté et l’auteur.
      2. Interface : Après chargement des signatures, les niveaux sont affichés et listés par ordre de difficulté. Les informations chargées à l’initialisation interface seront aussi affichées pour chacun des niveaux.
      3. Evènement : Lors du clique sur un bouton de niveau, la page conception sera affichée avec les paramètres Didacticiel et niveaux.
2. Conception
   1. Généralité
      1. L’interface : Elle est composée d’une grille en 2 dimensions ; lignes blanches sur fond bleu foncé. Sur la droite, un menu propose la liste des différents composants électriques (cf 3. Modules). En haut de l’écran, se trouve la barre d’outils (cf 4. Barre d’outils).
      2. Evènements :
         1. Le clique gauche de la souris sur l’interface permet d’ajouter le module sélectionné dans la liste des composants électriques à la position du curseur de la souris. Si un module existe déjà à l’emplacement du curseur, alors le menu propriété s’ouvrira en bas de l’écran (cf 5. Propriétés)
         2. Le clique droit de la souris sur l’interface permet de supprimer le composant sous le curseur de la souris.
         3. Le clique gauche sur un module de la liste des composants permet de le sélectionner
         4. Le clique gauche sur un élément de la barre d’outil permet de réaliser des actions (cf 4. Barre d’outils/Evènement)
         5. Le scrolling de la souris permet de zoomer/dézoomer la grille.
   2. Niveau
      1. Chargement : Le fichier niveau (.lvl) sera chargé depuis le paramètre envoyé selon un certain format afin de pouvoir détailler les instructions du niveau, les textes affichés, les schémas et toutes informations nécessaires. Les fichiers niveaux seront ainsi facilement manipulables car extérieur au code C.
      2. Interface : Un score en haut à droite sera affiché et indiquera à l’utilisateur son pourcentage d’avancement au sein du niveau.
      3. Avancement : Le niveau est décomposé en étape(instructions ou informations). Pour passer chaque étape, l’utilisateur devra suivre les instructions demandés et les réaliser : cela peut-être un module à placer ou bien appuyer sur la touche espace après une information ou remarque. Le score augmentera.
      4. Fin : Une fois le niveau terminé, l’utilisateur passera en mode Libre sans suppression du circuit électrique.
3. Module
   1. Initialisation : Liste des modules : Pile, résistance, DEL, diode, transistor (NPN & PNP), condensateur et inductance.
      1. Icones et textes : Les icones de chaque module sera chargé depuis le dossier image et afficher sur l’interface des modules avec le nom du module.
   2. Fonction : une fois placée sur la grille, le composant sera associé à une fonction qui calculera la tension et le courant de sortie selon la tension et le courant d’entrée ainsi que les propriétés associés au composant.
4. Barre d’outils (non définitif)
   1. Initialisation : Liste des outils : Nouveau circuit, tester le circuit, exporter en fichier image et quitter
   2. Nouveau circuit : supprime le circuit actuel pour en créer un nouveau
   3. Tester le circuit : active le circuit et calcule les tensions et courant dans chaque partie du circuit. Sur le schéma, la couleur vert-rouge représentera la tension et les points en violet, le courant
   4. Exporter en fichier image : enregistre dans un fichier le circuit actuel dans un format image
5. Propriété (non définitif)
   1. Initialisation :
      1. Evènement de création : Le panneau des propriété sera affiché quand l’utilisateur cliquera sur un module placé sur la grille. A ce moment-là, on charge les différents propriétés modifiables de l’objet. Par exemple, on pourra changer la résistance (en ohm) d’une résistance ou bien la capacité d’un transistor.
      2. Modulaire : Chaque propriété affichée sera modulable selon la nature du composant.

# Organigramme simplifié



# Suivi :

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | 05-nov | Kernel Week | 12-nov | Interface Week | 19-nov | Event Week | 26-nov | 03-déc |
| Sophie | Etude de SDL  Développement partie SDL de l’appli  Fonctionnalité 1.a.i | 1.b.i  2.a.ii.(5) | 2.b.ii.(1) |  |  |  |  |  |
| Alan | Etude de SDL : OK  Développement partie SDL : OK  Fonctionnalité 2.a.i : OK | 3.a.i  4.a |  |  |  |  |  |  |

|  |  |  |  |
| --- | --- | --- | --- |
|  | 10-déc | 17-déc | 24-déc |
| Sophie |  |  |  |
| Alan |  |  |  |

Note : trouver un créneau hebdomadaire (~1h30) afin de mettre en commun ce qui a été fait et ce qui sera fait.

Remarques :

* Ne pas utiliser des variables globales -> préféré les structures de programme
* Tout doit être organisé selon le noyau de l’app

# Documentation noyau

Créer une nouvelle interface (sans événement)

* Créer un nouveau fichier .c et .h (nomenclature : iXxxx.c où Xxxx est le nom de l’interface)
* Fonctions à créer :
  + initLayout\_xxxx(Renderer, Surface) -> Fonction qui initialise des données propres au layout
  + update\_xxxx(Window, Renderer, Surface) -> Fonction appellé à chaque réinitilisation de la fenetre (pour le dessin de la page)
* Ajouter l’appel de initLayout à la fin de postInit() du kernel
* Ajouter l’appel de update au niveau du commentaire de updateApp du kernel