



Adjivas Brezaire Flime Jpepin

Table des matières

1	Premiere partie	2
1.1	Préambule	2
1.2	Introduction	3
1.2.1	Utilisation	3
1.2.1.1	Programme <code>neko -CLI-</code>	4
1.2.1.2	Commande <code>neko -builtin-</code>	4
1.2.2	Programmeur	4
1.2.2.1	Fonctions dynamiques <code>-FFI-</code>	5
1.3	Editeur	6
1.3.1	Programme	6
1.3.1.1	Interface	6
1.3.1.2	Raccourcis clavier	7
1.3.2	Bibliothèque	8

Chapitre 1

Premiere partie

1.1 Préambule



[Wikipetan](#) -ウィキペたん-

Une nékoe -ねこみみ- est un persona d' animé japonais avec des traits de chat -mimikko¹ -.

Le GlyphArt est l' écriture d' une image via des caractères compris dans l' Unicode privé, ce projet démontre ce procédé via [Image2font](#).

L' [SVG OpenType](#) est un format ouvert de police de caractères vectorielles multicolors.

Arcana Azurea Pitou est une programmeuse nékoe fictive de terminal inventée pour assister des utilisateurs.

1. Kemonomimi ou mimikko est un personnage humain d' animé avec les caractéristiques d' un animal telles que sa personnalité ou encore son physique -獣耳-.

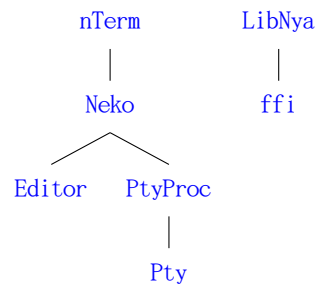
1.2 Introduction

Arcana Azurea Pitou est une nékoe de terminal qui a pour but d'apporter les Arts, la Culture et une assistance à qui saura utiliser un shell. Humanisée d'émotions et fondée sur l'expérience de la [chambre chinoise](#), celle-ci sera donc instruite via des bibliothèques.

L'organisation GitHub Arukana de philosophie [libriste](#) fut créée pour distribuer et maintenir communautairement les dépôts nécessaires au développement de cette Nékoe de terminal.

Book	La documentation du projet neko.
PtyProc	L' intergiciel -Middleware- et l' arrière-plan -back-end- de l' émulateur du terminal VT100 .
Editor	L' éditeur et la bibliothèque d' expression de la nékoe.
Neko	Le programme neko.
LibNya	La bibliothèque dynamique de teste du programme neko.
ffi	L' en-têtes - header - sont les déclarations des structures et énumérations de la nékoe.
Image2font	Le convertisseur d' images en une police d' écriture.
nTerm	L' interface graphique de émulateur.

Ces dépôt sont interdépendant tel que :



1.2.1 Utilisation

La constante d' environnement `$NEKO_PATH` pourra être définie à « `$HOME/.neko` » et comprendra les sous-répertoires `lib`, `rep`, `texels` et `sprites`. Sinon, les [bibliothèques](#) de l' organisation Arukana ceux reporterons à la constante `$CARGO_MANIFEST_DIR` de valeurs relatives aux répertoires contenant les [fichiers manifests](#).

sprite	Les sprites de la nékoe.
texels	Les texels de la nékoe.
lib	Les sources des progiciels.
rep	Les bibliothèques dynamiques des progiciels.

1.2.1.1 Programme neko -CLI-

<code>--help</code>	Imprime ce menu d' aide
<code>--version, -V</code>	Imprime la version du programme neko
<code>--command, -c</code> <code>[/bin/zsh]</code>	Précise le processus fils.
<code>--repeat, -r</code> <code>[1000]</code>	Précise le temps de répétition du maintien d' une touche enfoncée tel que $\{2, \dots, N\}$.
<code>--interval, -i</code> <code>[1000]</code>	Précise le temps d' intervalle durant les répétitions tel que $\sum_{i=repeat}^{\infty} U_{interval} \times i$.

Fonctionnalité supplémentaire « keyboard-time ».

1.2.1.2 Commande neko -builtin-

Le programme neko substitura la commande neko(1) pour son processus enfant uniquement et qui sera à l' occurrence notre [interpréteur de commandes](#).

Cette commande comprend les options si-suivantes :

<code>install</code> <code><url></code>	<code>https://git...</code>	Installe depuis dépôt git un plugiciel.
<code>uninstall</code> <code><author@libname></code>	<code>Arukana@LibNya</code>	Désinstalle les sources et la bibliothèque dynamique d' un plugiciel - plugin -.
<code>mount</code> <code><author@libname></code> <code>[<priority>]</code>	<code>Arukana@LibNya</code> <code>1</code>	Monte une bibliothèque dynamique avec une priorité de niveau zero.
<code>unmount</code> <code><author@libname></code>	<code>Arukana@LibNya</code>	Démonte une bibliothèque dynamique.
<code>update</code> <code><author@libname></code>	<code>Arukana@LibNya</code>	Révisé une bibliothèque dynamique.

« install » implicite « mount ».

« uninstall » implicite « unmount ».

1.2.2 Programmeur

Les sources d' une bibliothèque dynamique devront toujours comprendre :

Un Makefile devra compiler une bibliothèque nommé « `author@repository.dylib` » via la règle default.

Un Manifest nommé « `Neko.toml` » -[TOML](#)- et contenant les attributs facultatifs ci-suivants :

<code>priority = 0</code>	De type Integer compris entre $\{-(2^{64} - 1), \dots, 2^{63} - 1\}$ (Voir la spécification ISO/IEC 9899 TC3 ; § 5.2.4.2.1).
<code>[dependencies.name]</code>	
<code>git = "https://git..."</code>	De type Inline Table et décrivant une dépendence.

1.2.2.1 Fonctions dynamiques -FFI-

Le bibliothèque LibNya est pour ses branches C et Rust un exemple d' utilisation de la liste des fonctions si-suivantes :

void install (t_lbstat *lib, void **data)	Quand la bibliothèque est installée.
void uninstall (t_lbstat *lib, void **data)	Quand la bibliothèque va être désinstallé.
void mount (t_lbstat *lib, void **data)	Quand la bibliothèque est montée.
void unmount (t_lbstat *lib, void **data)	Quand la la bibliothèque est démontée.
void idle (t_lbstat *lib, void **data)	Pour chaque cycle compris entre chaques événements.
void process (t_lbstat *lib, void **data, char *name, pid_t pid)	Quand le processus courant change.
void command (t_lbstat *lib, void **data, char *lined)	Quand une ligne de commande va être saisie.
void key_unicode_down (t_lbstat *lib, void **data, unsigned long long key)	Quand une touche enfoncée va être envoyée.
void key_string_down (t_lbstat *lib, void **data, char *copy)	Quand un texte va être envoyé.
void key_repeat_down (t_lbstat *lib, void **data, unsigned int repeat)	Quand une touche est maintenue enfoncée {2...N}.
void key_interval_down (t_lbstat *lib, void **data, long long repeat)	Quand KeyDownRepeat, donne l' intervalle : $\sum_{i=repeat}^{\infty} U_{interval} \times i$.
void mouse_pressed (t_lbstat *lib, void **data, t_mouse code, unsigned short cartesian[2])	Quand le curseur va être pressé.
void mouse_released (t_lbstat *lib, void **data, t_mouse code, unsigned short cartesian[2])	Quand le curseur va être relâché.
void input (t_lbstat *lib, void **data, char *output)	Quand un texte va être imprimé sur l' entrée standard du processus fils.
void output (t_lbstat *lib, void **data, char *output)	Quand un texte est imprimé depuis la sortie standard du processus fils.
void resized (t_lbstat *lib, void **data, Winszed *win)	Quand la taille de la fenêtre change.

■ Fonctionnalité supplémentaire « keyboard-time ».

1.3 Editeur

L' Editeur est à la fois :

1. Un programme qui selon une liste de primitive permettra de deviner des expressions en commandes. Interface utilisateur en mode texte `-TUI-`.
2. Une bibliothèque et dépendence du projet Neko, qui chargera un dictionnaire de `sprite`.

1.3.1 Programme

L' Editeur comprend une liste de sprite, chacun de 1 à 16 `-SPEC_MAX_DRAW-` dessins ; un dessin comprenant 10×5 `texels` `-SPEC_MAX_XY-`. Ainsi, par `arrangement avec répétition`, nous pourrons définir pour 5 `emotions` `-SPEC_MAX_EMOTION-` notre nombre limite expressions à :

$$\overline{A_5^{(16 \times 10 \times 5)}} = (16 \times 10 \times 5)^5 = 1Y3.27680054e14] \quad (1.1)$$

1.3.1.1 Interface

L' interface est représentée par :

Un menu de commandes.

Une liste déroulante de sprite.



Le numéro, délais en milliseconde et posture d' un dessin.

Le dessin par caractère `-PUA2-`, parti du corp et `emotion`.

La liste des `emotions` disponible pour la case courrante.

La somme de toute les `emotions` non nul par posture pour la `sprite` courrante.

Tel que :





























```
Quit <q>
bust
0 - PT0.200S : Talk

HeHeHe _____
HeHeHe _____
      Mo _____
      _____
      _____
o_
1 - PT0.200S : NotTalk

HeHeHe _____ ooo _____
HeHeHe _____ ooo _____
      Mo _____
      _____
      _____
_o
-Talk -NotTalk Heart :Shocked
```

La représentation non nul « `-Talk -NotTalk Heart :Shocked` » pourra être utilisé via les énumérations `Part` et `Emotion` d' écrite par la bibliothèque `ffi` pour reproduire la même expression depuis une bibliothèque dynamique.

2. Caractère privé compris entre $\{U+E000, \dots, U+F8FF\}$ (Voir la spécification [The Unicode Standard Version 9.0](#) ; Chapt. Properties §3.5 – Private Use.

1.3.1.2 Raccourcis clavier

La saisie est adapté selon la disposition des touches du [terminal ADM-3A](#) de la société [Lear Siegler](#).

Touche	Description
 +  ou 	Quitter le programme.
 +  ou 	Copie la commande dans le presse-papier.
 ou 	Sélectionne la première animation.
 ou 	Sélectionne l' animation précédente.
 ou 	Sélectionne l' animation suivante.
 ou 	Sélectionne la dernière animation.
 ou 	Sélectionne le sprite précédent.
 ou 	Sélectionne le sprite suivant.
 ou 	Sélectionne la cellule de gauche.
 ou 	Sélectionne la cellule du haut.
 ou 	Sélectionne la cellule du bas.
 ou 	Sélectionne la cellule de droite.
 ... 	Change l' émotion du groupe de cellules courant.

■ Fonctionnalité supplémentaire « Clipboard ».

1.3.2 Bibliothèque

La fonction change l'etat d'une sprite parmi une liste ...

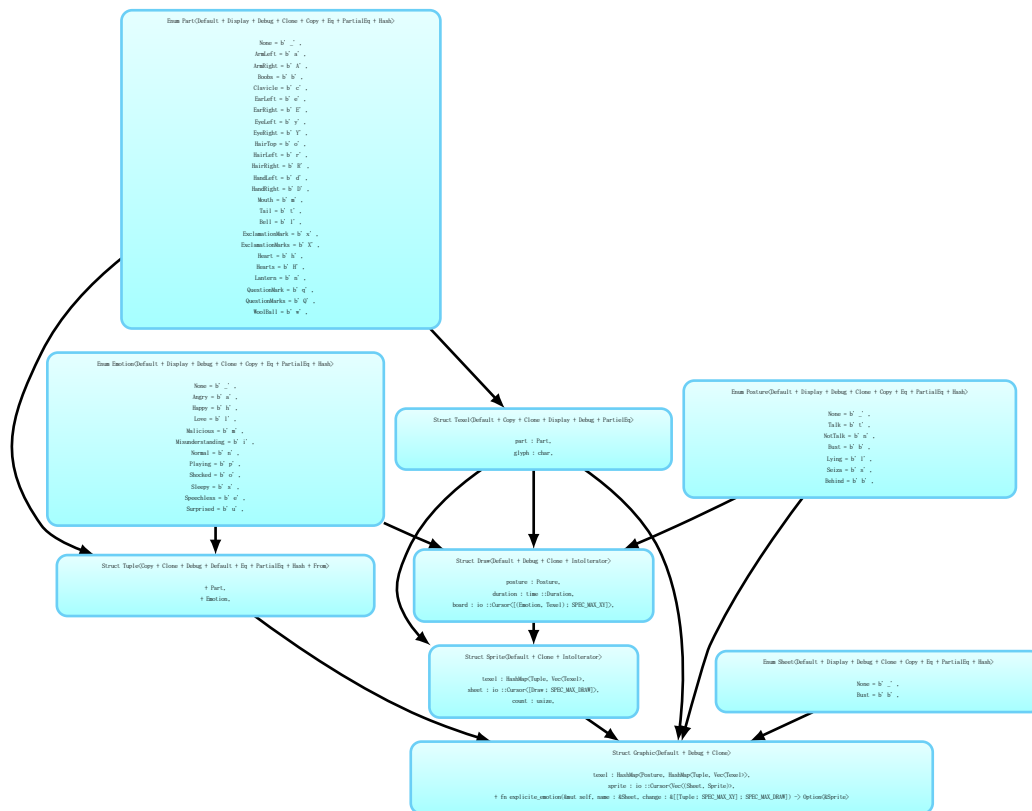


Diagramme UML³ simplifié du module graphique.

3. En génie logiciel, le langage de modélisation orienté objet unifié de l'anglais « Unified Modeling Language » - UML - est la représentation schématique d'un programme par de l'orienté objet