Nyle　minimal graphics framework using Ruby/GTK3 and Cairo

1. Overview

(1)Introduction (はじめに)

Ruby でゲームの作成やグラフィックを用いたプログラミングをするために、Ruby/GTK3 と Cairo をベースにした簡単なフレームワークを作成しました。
既存の諸々のグラフィックライブラリや Processing といった他言語のライブラリ構成などを参考にして、Ruby で手軽にグラフィック処理をおこなえるように設計しました。

(2)Installation (導入方法)

下記のように nyle をインストールしてください。
依存パッケージの Ruby/GTK3 も一緒にインストールされます。

```
gem install nyle
```

(3)Usage (使用方法)

Nyle を使用する際には、プログラムの冒頭に下記のコードを記述して requre してください。

```
require 'nyle'
```

(4)Samples(サンプルプログラム)

nyle.rb をインストールすると、インストール先の samples サブディレクトリにいくつかのサンプルプログラムが配置されます。
samples サブディレクトリに移動し、下記のように各サンプルプログラムを実行してください。

```
ruby nyle_basics.rb
```

(5)Restrictions (制約事項など)

[動作環境]

OS	Ruby	GTK3	Cairo
Windows10	2.4 以降	3.3.0 以降	1.16.2 以降
MacOS X	2.4 以降	3.3.0 以降	1.16.2 以降
Linux	2.4 以降	3.3.0 以降	1.16.2 以降

[注意事項]

・上表に記載されているバージョンより古いものでも動作は可能ですが、画面表示等になんらかの支障が生じる場合があります。
・一定タイミングごとの呼び出しには Ruby/GTK3(GLib2)のタイマー機能を用いており、重い描画処理などを頻繁におこなうような場合はプログラムの遅延または停止が生じる可能性があります。また、キーボードやマウスによる入力イベントの取得についても、若干の取りこぼしが生じる可能性があります。

2. Coding Style

(1)Baic Coding Style (基本的なプログラミング方法)

下記のように Nyle::Screen から Screen クラスを作成(継承)し、その中の draw メソッドと update メソッドを実装(オーバーライド)してください。初期設定等に関する処理は、initialize メソッドに記述してください。

Screen クラスの draw メソッドと update メソッドは、Nyle のモジュールから一定のタイミング(基本的に 15ms ごと)で呼び出される仕組みになっていますので、明示的にループの処理を記述する必要はありません。

なお、Screen クラスのインスタンス生成と表示、および GTK3 のメインスレッド実行については、下記の末尾 2 行のように固定的な記述でプログラムができあがります。

```
require 'nyle'

class Screen < Nyle::Screen
  def initialize
    super       # スーパークラスの initialize を呼び出し
  end
  def draw
    # 描画用の処理など
  end
  def update
    # 座標計算の処理など
  end
end

Screen.new.show_all       # Screen クラスのインスタンスを生成して表示
Gtk.main                  # GTK3 のメインスレッドを実行
```

Nyle を用いたプログラミングの実例については、samples ディレクトリの各サンプルプログラムをご参照ください。

(2)Controlling multiple Screens (複数の画面の切替え制御)

複数の画面(シーン)を切替える必要のあるプログラムについては、Nyle::Frame クラスから派生させた Frame クラスのインスタンスを使って複数の Screen クラスのインスタンスを制御します。

まず、Frame クラスの initialize メソッドで各 Screen クラスのインスタンスを生成し、最初に表示させたい Screen クラスのインスタンスを set_current メソッドで指定します。

そして、状態遷移テーブルである @transition という名前の配列に、下記のように「表示中の Screen がどのようなステータスになったらどの Screen に遷移するか」という情報をハッシュの形式で必要な分だけ与えていきます。

```
@transition << {current: @screen_main, status: :CLEAR, next: @screen_sub}

  ※上記の場合
    「Frame に表示中の @screen_main のステータスが :CLEAR になったら
     Frame の表示を @screen_sub に切り替える」
   という定義になります。
```

あとは、Nyle のモジュールから一定のタイミング(基本的に 15ms ごと)で状態遷移テーブルの内容が参照され、指定されたステータスに応じて表示中の Screen クラスのインスタンスが自動的に切り替わります。

なお、各 Screen クラスのインスタンスは、Frame に表示される際に resume メソッドが呼び出され、Frame から表示されなくなる際に suspend メソッドが呼び出されます。
これらの 2 つのメソッドを適宜実装することによって、画面(シーン)の切り替わりに応じて各インスタンスの内部状態を制御することができます。

```
require 'nyle'

class ScreenMain < Nyle::Screen     # メインシーン
  def initialize
    super       # スーパークラスの initialize を呼び出し
  end
  def draw
    # 描画用の処理など
  end
  def update
    # 座標計算の処理など
  end
  def suspend
    @status = nil     # 表示対象でなくなる際の処理
  end
end

class ScreenSub1 < Nyle::Screen     # サブシーン１
  def initialize
    super       # スーパークラスの initialize を呼び出し
```

```
  end
  def draw
    # 描画用の処理など
  end
  def update
    # 座標計算の処理など
  end
  def resume
    @status = :ACTIVE    # 表示対象になる際の処理
  end
end

class ScreenSub2 < Nyle::Screen    # サブシーン２
  def initialize
    super       # スーパークラスの initialize を呼び出し
  end
  def draw
    # 描画用の処理など
  end
  def update
    # 座標計算の処理など
  end
  def resume
    @status = :ACTIVE    # 表示対象になる際の処理
  end
end


class Frame < Nyle::Frame
  def initialize
    super
    @screen_main = ScreenMain.new      # メインシーンの生成
    @screen_sub1 = ScreenSub1.new      # サブシーン１の生成
    @screen_sub2 = ScreenSub2.new      # サブシーン２の生成

    # 状態遷移テーブル
    # @screen_main のステータス(@status)が :CLEAR になったら @screen_sub1 に遷移
    # @screen_main のステータス(@status)が :OVER になったら @screen_sub2 に遷移
    @transition << {current: @screen_main,  status: :CLEAR, next: @screen_sub1}
    @transition << {current: @screen_main,  status: :OVER,  next: @screen_sub2}

    self.set_current(@screen_main)         # メインシーンを現在の表示対象に設定
  end
end

Frame.new.show_all      # Frame クラスのインスタンスを生成して表示
Gtk.main                # GTK3 のメインスレッドを実行
```

[画面遷移のイメージ図]

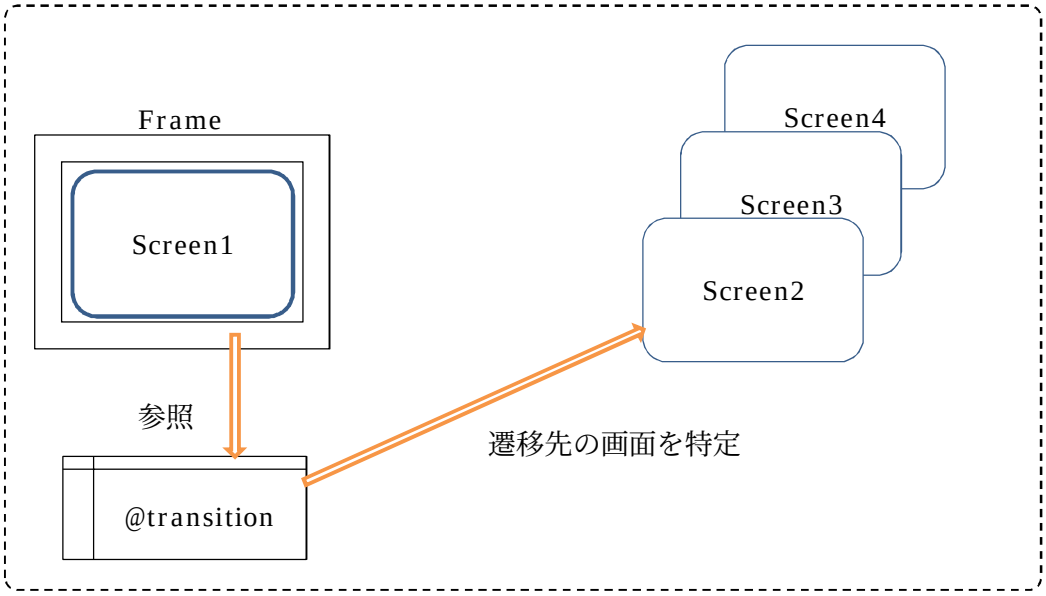

Fig.1 状態遷移テーブルの参照と遷移先画面の特定

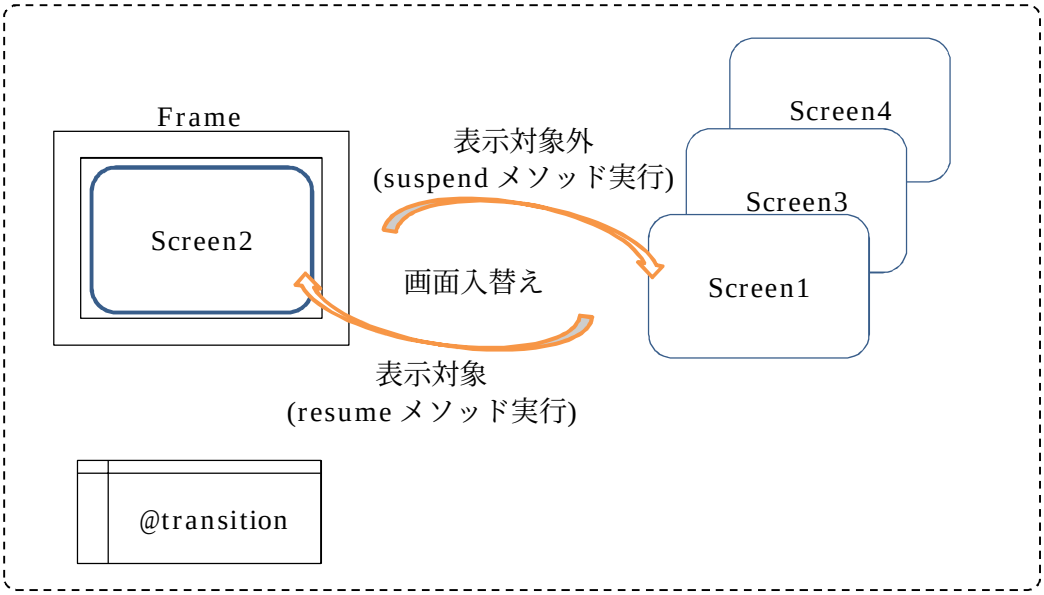

Fig.2 画面の入替えとメソッドの実行

3. API Summary

(1) Nyle モジュール内の描画系メソッド

	Methods	Description	Notes
1	Nyle.draw_line	直線の描画	
2	Nyle.draw_rect	矩形の描画	
3	Nyle.draw_circle	円の描画	
4	Nyle.draw_shape	頂点指定による図形の描画	
5	Nyle.draw_text	文字の表示	
6	Nyle.load_image	画像ファイルの読込み	
7	Nyle.load_image_tiles	画像ファイルの読込み	m×n 個のピースに分割
8	Nyle.draw_image	画像の表示	
9	Nyle.save_image	画像ファイルへの保存	
10	Nyle.pixel	指定座標の色情報を取得	
11	Nyle.pixel?	指定座標の色判定	
12	Nyle.save	現在の座標系の状態を保持	
13	Nyle.translate	座標系の移動	
14	Nyle.rotate	座標系の回転	
15	Nyle.scale	座標系の拡大/縮小	

(2) Nyle モジュール内の状態参照系メソッド

	Methods	Description	Notes
1	Nyle.mouse_x	マウスポインタの x 座標を取得	
2	Nyle.mouse_y	マウスポインタの y 座標を取得	
3	Nyle.mouse_down?	マウスボタンが押されているか	
4	Nyle.mouse_press?	マウスボタンが押された瞬間か	
5	Nyle.mouse_release?	マウスボタンが放された瞬間か	
6	Nyle.key_down?	キーが押されているか	
7	Nyle.key_press?	キーが押された瞬間か	
8	Nyle.key_release?	キーが放された瞬間か	
9	Nyle.mask_control?	[Ctrl]キーが押されているか	
10	Nyle.mask_shift?	[Shift]キーが押されているか	
11	Nyle.screen_width	表示中のスクリーンの幅を取得	
12	Nyle.screen_height	表示中のスクリーンの高さを取得	

(3)Nyle モジュール内のその他メソッド

	Methods	Description	Notes
1	Nyle.cr	CairoContext の取得	
2	Nyle.quit	プログラムの終了	

(4)Nyle モジュール内のクラス

	Classes	Description	Notes
1	Nyle::Frame	フレームクラス	Nyle::Screen 制御用
2	Nyle::Screen	スクリーンクラス	

(5)Nyle モジュールから参照する環境変数

	Variables	Description	Notes
1	NYLE_INTERVAL	画面描画の時間間隔	5〜60 (milli seconds)

4. API Reference

(1)Nyle モジュール内の描画系メソッド

(1-1) Nyle.draw_line

synopsis	直線の描画 [書式] Nyle.draw_line(x1, y1, x2, y2, {options})		
arguments	x1	始点の x 座標	
	y1	始点の y 座標	
	x2	終点の x 座標	
	y2	終点の y 座標	
options	weight	線の太さ	1 以上の整数値
		(デフォルト値)	2
	color	線の色	Appendix(3)参照
		(デフォルト値)	:BLACK
	a	線の色のα値	0.0～1.0
		(デフォルト値)	1.0
	cap	線端の形状	:BUTT 通常 :ROUND 丸み :SQUARE 角
		(デフォルト値)	:BUTT
e.g.	Nyle.draw_line(10, 10, 110, 110) Nyle.draw_line(20, 10, 120, 120, {weight: 5}) Nyle.draw_line(30, 10, 130, 130, {cap: :ROUND}) Nyle.draw_line(40, 10, 140, 140, {color: :BLUE}) Nyle.draw_line(50, 10, 150, 150, {color: :RED, weight: 3})		

(1-2) Nyle.draw_rect

synopsis	矩形の描画 [書式] Nyle.draw_rect(x, y, width, height, {options})			
arguments	x	矩形の左上 x 座標		
	y	矩形の左上 y 座標		
	width	矩形の幅		
	height	矩形の高さ		
options	weight	線の太さ	1 以上の整数値	
		(デフォルト値)	2	
	color	線の色	Appendix(3)参照	
		(デフォルト値)	:BLACK	
	a	線の色のα値	0.0～1.0	
		(デフォルト値)	1.0	
	fill	塗りつぶし	true false	塗りつぶす 塗りつぶさない
		(デフォルト値)	false	
	round	角の丸み	0 以上の整数値	
		(デフォルト値)	0	
e.g.	Nyle.draw_rect(10, 10, 60, 30) Nyle.draw_rect(20, 20, 60, 30, {weight: 5}) Nyle.draw_rect(30, 30, 60, 30, {weight: 5, color: :ORANGE}) Nyle.draw_rect(40, 40, 60, 30, {color: :VIRIDIAN, fill: true}) Nyle.draw_rect(50, 50, 60, 30, {weight: 5, color: :ORANGE, round: 5}) Nyle.draw_rect(60, 60, 60, 30, {color: :VIRIDIAN, fill: true, round: 5})			

(1-3) Nyle.draw_circle

synopsis	円の描画 [書式] Nyle.draw_circle(x, y, r, {options})		
arguments	x	円の中心 x 座標	
	y	円の中心 y 座標	
	r	円の半径	
options	weight	線の太さ	1 以上の整数値
		(デフォルト値)	2
	color	線の色	Appendix(3)参照
		(デフォルト値)	:BLACK
	a	線の色のα値	0.0～1.0
		(デフォルト値)	1.0
	fill	塗りつぶし	true する false しない
		(デフォルト値)	false
e.g.	Nyle.draw_circle(170, 150, 20) Nyle.draw_circle(170, 150, 30, {weight: 5}) Nyle.draw_circle(270, 150, 30, {weight: 5, color: :ORANGE}) Nyle.draw_circle(370, 150, 30, {color: :VIRIDIAN, fill: true})		

(1-4) Nyle.draw_shape

synopsis	頂点指定による図形の描画 [書式] Nyle.draw_shape(points, {options})			
arguments	points	頂点座標[x, y]の配列		
options	weight	線の太さ	1 以上の整数値	
		(デフォルト値)	2	
	color	線の色	Appendix(3)参照	
		(デフォルト値)	:BLACK	
	a	線の色のα値	0.0～1.0	
		(デフォルト値)	1.0	
	fill	塗りつぶし	true false	塗りつぶす 塗りつぶさない
		(デフォルト値)	false	
	close	閉鎖図形 (始点と終点の結合)	true false	閉鎖図形にする 閉鎖図形にしない
		(デフォルト値)	false	fill: true の場合は 自動的に始点と 終点が結合される
	cap	線端の形状 (始点と終点のみ)	:BUTT :ROUND :SQUARE	通常 丸み 角
		(デフォルト値)	:BUTT	
e.g.	Nyle.draw_shape([[10, 10], [100, 50], [50, 100]]) Nyle.draw_shape([[10, 10], [100, 50], [50, 100]], {close: true}) Nyle.draw_shape([[10, 10], [100, 20], [120, 100], [30, 130]], {color: :BLUE, fill: true})			

(1-5) Nyle.draw_text

synopsis	文字列の表示 [書式] Nyle.draw_text(x, y, text, {options})			
arguments	x	x 座標		
	y	y 座標		
	text	文字列		
options	size	フォントサイズ		
		(デフォルト値)	32	
	color	文字の色	Appendix(3)参照	
		(デフォルト値)	:BLACK	
	a	文字の色のα値	0.0～1.0	
		(デフォルト値)	1.0	
e.g.	Nyle.draw_text(40, 100, "【Nyle ナイル】") Nyle.draw_text(40, 150, "October 2018", {color: :RED, size: 24})			

(1-6) Nyle.load_image

synopsis	画像ファイルの読込み [書式] image = Nyle.load_image(filename, {options}) [戻値] GdkPixbuf::Pixbuf クラスのインスタンス		
arguments	filename	ファイル名	対応フォーマット png, jpg, gif, bmp
options	color_key	透過色	Appendix(3)参照
		(デフォルト値)	nil
	sx	拡大率(横方向)	0.1 以上の値
		(デフォルト値)	1.0
	sy	拡大率(縦方向)	0.1 以上の値
		(デフォルト値)	1.0
	cx	抽出範囲の左上 x 座標	
		(デフォルト値)	nil
	cy	抽出範囲の左上 y 座標	
		(デフォルト値)	nil
	cw	抽出範囲の幅	
		(デフォルト値)	nil
	ch	抽出範囲の高さ	
		(デフォルト値)	nil
e.g.	image = Nyle.load_image("picture.png", {sx: 0.7, sy: 0.7, color_key: :WHITE}) image = Nyle.load_image("picture.png", {cx: 10, cy: 10, cw: 100, ch: 100})		

(1-7) Nyle.load_image_tiles

synopsis	画像ファイルの読込み(m×n 個のピースに分割) [書式] image = Nyle.load_image_tiles(filename, xcount, ycount, {options}) [戻値] GdkPixbuf::Pixbuf クラスのインスタンスを収録した２次元配列		
arguments	filename	ファイル名	対応フォーマット png, jpg, gif, bmp
	xcount	横方向の分割数	
	ycount	縦方向の分割数	
options	color_key	透過色	Appendix(3)参照
		(デフォルト値)	nil
	sx	拡大率(横方向)	0.1 以上の値
		(デフォルト値)	1.0
	sy	拡大率(縦方向)	0.1 以上の値
		(デフォルト値)	1.0
e.g.	tiles = Nyle.load_image_tiles("picture.png", 3, 4) tiles = Nyle.load_image_tiles("picture.png", 3, 4, {sx: 1.2, sy: 1.2})		

(1-8) Nyle.draw_image

synopsis	画像の表示 [書式] Nyle.draw_image(x, y, image)	
arguments	x　　x 座標	
	y　　y 座標	
	image　　GdkPixbuf::Pixbuf クラスのインスタンス	
options	-	
e.g.	Nyle.draw_image(10, 10, image) Nyle.draw_image(150, 200, tiles[1][2])	

(1-9) Nyle.save_image

synopsis	画像ファイルへの保存 [書式] Nyle.save_image(filename)	
arguments	filename　　ファイル名	対応フォーマット png のみ
options	-	
e.g.	Nyle.save_image("capture.png")　# 表示中の画面を capture.png に保存	

(1-10) Nyle.pixel

synopsis	指定座標の色情報を取得 [書式] Nyle.pixel(x, y)	
arguments	x　　x 座標	
	y　　y 座標	
return	色情報 (#RRGGBBAA 形式の文字列)	Appendix(3)参照
e.g.	c = Nyle.pixel(100, 100)　# 座標(100, 100)の色情報を取得	

(1-11) Nyle.pixel?

synopsis	指定座標の色判定 [書式] Nyle.pixel?(x, y, color)	
arguments	x　　x 座標	
	y　　y 座標	
	color　　色	Appendix(3)参照
return	true　... 座標(x, y)の色が color と同じである false　... 座標(x, y)の色が color と同じではない	
e.g.	Nyle.translate(50, 50)	

(1-12) Nyle.save

synopsis	現在の座標系の状態を保持 [書式] Nyle.save do ... end [補足] do～end(または {～})で指定されたブロックを抜けるまでの間、 指定した座標系の状態が保持されます。 座標変換の有効範囲を限定したり、座標変換を再帰的におこなう場合に 使用してください。
e.g.	Nyle.save do Nyle.translate(50, 50) Nyle. rotate(Math::PI / 4) Nyle.scale(0.5, 0.5) Nyle.draw_line(10, 10, 100, 100) end

(1-13) Nyle.translate

synopsis	座標系の移動 [書式] Nyle.translate(tx, ty)	
arguments	tx　　x 方向の移動量	
	ty　　y 方向の移動量	
e.g.	Nyle.translate(50, 50)	

(1-14) Nyle.rotate

synopsis	座標系の回転 [書式] Nyle.translate(angle)	
arguments	angle　　回転角	radian
e.g.	Nyle. rotate(Math::PI / 4)	

(1-15) Nyle.scale

synopsis	座標系の拡大/縮小 [書式] Nyle.scale(sx, sy)	
arguments	sx　　x 方向の拡大率	0.1 以上の値
	sy　　y 方向の拡大率	0.1 以上の値
e.g.	Nyle.scale(0.5, 0.5)	

(2)Nyle モジュール内の状態参照系メソッド

(2-1) Nyle.mouse_x

synopsis	マウスポインタの x 座標を取得 [書式] Nyle.mouse_x
arguments	なし
return	x 座標(pixels)
e.g.	puts Nyle.mouse_x

(2-2) Nyle.mouse_y

synopsis	マウスポインタの y 座標を取得 [書式] Nyle.mouse_y
arguments	なし
return	y 座標(pixels)
e.g.	puts Nyle.mouse_y

(2-3) Nyle.mouse_down?

synopsis	マウスボタンが押されているかどうかの状態判定 [書式] Nyle.mouse_down?(button)
arguments	button MOUSE_L ... 左ボタン MOUSE_M ... 中ボタン MOUSE_R ... 右ボタン
return	true ... 押されている false ... 押されていない
e.g.	puts Nyle.mouse_down?(MOUSE_L) # 左ボタンが押されていれば true puts Nyle.mouse_down?(MOUSE_M) # 中ボタンが押されていれば true puts Nyle.mouse_down?(MOUSE_R) # 右ボタンが押されていれば true

(2-4) Nyle.mouse_press?

synopsis	マウスボタンが押された瞬間かどうかの状態判定 [書式] Nyle.mouse_press?(button)
arguments	button　MOUSE_L ... 左ボタン MOUSE_M ... 中ボタン MOUSE_R ... 右ボタン
return	true　... 押された瞬間である false　... 押された瞬間ではない
e.g.	puts Nyle.mouse_press?(MOUSE_L)　# 左ボタンが押された瞬間ならば true puts Nyle.mouse_press?(MOUSE_M)　# 中ボタンが押された瞬間ならば true puts Nyle.mouse_press?(MOUSE_R)　# 右ボタンが押された瞬間ならば true

(2-5) Nyle.mouse_release?

synopsis	マウスボタンが放された瞬間かどうかの状態判定 [書式] Nyle.mouse_release?(button)
arguments	button　MOUSE_L ... 左ボタン MOUSE_M ... 中ボタン MOUSE_R ... 右ボタン
return	true　... 放された瞬間である false　... 放された瞬間ではない
e.g.	puts Nyle.mouse_release?(MOUSE_L) #　左ボタンが放された瞬間ならば true puts Nyle.mouse_release?(MOUSE_M) # 中ボタンが放された瞬間ならば true puts Nyle.mouse_release?(MOUSE_R) #　右ボタンが放された瞬間ならば true

(2-6) Nyle.key_down?

synopsis	キーが押されているかどうかの状態判定 [書式] Nyle.key_down?(keyval)
arguments	キーコード (Appendix(2)参照)
return	true ... 押されている false ... 押されていない
e.g.	puts Nyle.key_down?(KEY_a) # [a]キーが押されていれば true puts Nyle.key_down?(KEY_Up) # [↑]キーが押されていれば true

(2-7) Nyle.key_press?

synopsis	キーが押された瞬間かどうかの状態判定 [書式] Nyle.key_press?(keyval)
arguments	キーコード (Appendix(2)参照)
return	true ... 押された瞬間である false ... 押された瞬間ではない
e.g.	puts Nyle.key_press?(KEY_a) # [a]キーが押された瞬間であれば true puts Nyle.key_press?(KEY_Up) # [↑]キーが押された瞬間であれば true

(2-8) Nyle.key_release?

synopsis	キーが放された瞬間かどうかの状態判定 [書式] Nyle.key_release?(keyval)
arguments	キーコード (Appendix(2)参照)
return	true ... 放された瞬間である false ... 放された瞬間ではない
e.g.	puts Nyle.key_release?(KEY_a) # [a]キーが放された瞬間であれば true puts Nyle.key_release?(KEY_Up) # [↑]キーが放された瞬間であれば true

(2-9) Nyle.mask_control?

synopsis	[Ctrl]キーが押されているかどうかの状態判定 [書式] Nyle.mask_control?
arguments	なし
return	true ... 押されている false ... 押されていない
e.g.	puts Nyle.mask_control? # 左右いずれかの[Ctrl]キーが押されていれば true

(2-10) Nyle.mask_shift?

synopsis	[Shift]キーが押されているかどうかの状態判定 [書式] Nyle.mask_shift?
arguments	なし
return	true ... 押されている false ... 押されていない
e.g.	puts Nyle.mask_shift? # 左右いずれかの[Shift]キーが押されていれば true

(2-11) Nyle.screen_width

synopsis	表示中のスクリーンの幅を取得 [書式] Nyle.screen_width
arguments	なし
return	幅(pixels)
e.g.	puts Nyle.screen_width

(2-12) Nyle. screen_height

synopsis	表示中のスクリーンの高さを取得 [書式] Nyle.screen_height
arguments	なし
return	高さ(pixels)
e.g.	puts Nyle.screen_height

(3)Nyle モジュール内のその他メソッド

(3-1) Nyle.cr

synopsis	CairoContext の取得 [書式] Nyle.cr [備考] 'Nyle'は CairoContext に関するメソッドのうち 最小限のものしかラッピングしていないため、 それ以外のメソッドについては Nyle.cr で CairoContext を取得して 必要な処理を記述してください。 (詳細は Cairo についてのドキュメントを参照してください)
arguments	なし
return	現在表示中の画面の CairoContext
e.g.	Nyle.cr.arc(100, 100, 30, 0.0, Math::PI) Nyle.clip

(3-2) Nyle.quit

synopsis	プログラムの終了 [書式] Nyle.quit
arguments	なし
return	なし
e.g.	Nyle.quit　# GTK3 のメインスレッドを終了

(4)Nyle モジュール内のクラス

(4-1) Nyle::Frame

description	フレームクラス スクリーンを制御するための表示用ウィンドウ
super class	Gtk::Window
constructor	new(width, heigh, {options}) [arguments] width　フレームの幅 (デフォルト 640) height　フレームの高さ (デフォルト 480) [options] title　フレームのタイトル (デフォルト 'Nyle') [e.g.] Nyle::Frame.new Nyle::Frame.new(400, 300) Nyle::Frame.new(400, 300, {title: 'Sample'}) Nyle::Frame.new({title: 'Sample'})
class methods	-
instance methods	set_current(screen) フレームに表示させるスクリーンをセットする [arguments] screen　フレームに表示させる Screen クラスのインスタンス

(4-2) Nyle::Screen

description	スクリーンクラス 描画用の画面
super class	Gtk::DrawingArea
constructor	new(width, heigh, {options}) [arguments] width　スクリーンの幅 (デフォルト 640) height　スクリーンの高さ (デフォルト 480) [options] bgcolor　スクリーンの背景色 (デフォルト :WHITE) trace　スクリーンの描画軌跡属性 (デフォルト false) true: 描画軌跡あり false: 描画軌跡なし [e.g.] Nyle::Screen.new Nyle::Screen.new(400, 300) Nyle::Screen.new(400, 300, {bgcolor: :BLACK}) Nyle::Screen.new(400, 300, {trace: true}) Nyle::Screen.new({bgcolor: :BLACK, trace: true})
class methods	-
instance methods	show_all(title) フレームを生成してスクリーンを表示する [argument] title　フレームのタイトル (デフォルト 'Nyle')
abstract method	draw フレームから定期的に呼び出される描画用メソッド (サブクラスで中身を実装)
	update フレームから定期的に呼び出される更新用メソッド (サブクラスで中身を実装)
	suspend フレームから外れる際に実行されるメソッド (サブクラス内で中身を実装)
	resume フレームにセットされる際に実行されるメソッド (サブクラス内で中身を実装)

(5)Nyle モジュールから参照する環境変数

(5-1) NYLE_INTERVAL

synopsis	画面描画の時間間隔を調整するための環境変数 [設定例] export NYLE_INTERVAL=20 # bash (MacOS X / Linux) SET NYLE_INTERVAL=20 # cmd (Windows) $env:NYLE_INTERVAL=20 # PowerShell (Windows) [備考] 設定値は 10～60 の間で指定してください。 環境変数 NYLE_INTERVAL が未設定の場合、時間間隔は 15(ms)が 適用されます。 [備考] 環境などによって画面描画の時間間隔が不安定になる場合があるため、 値を調整できるように環境変数を設けました。

Appendix

(1)MouseButton indexes

Index	Description	Notes
MOUSE_L	左ボタン	1
MOUSE_M	中ボタン	2
MOUSE_R	右ボタン	3

(2)Key codes

Name	Description	Notes
KEY_space	スペース	0x020
KEY_BackSpace	バックスペース	0xff08
KEY_Tab	タブ	0xff09
KEY_Return	エンター	0xff0d
KEY_Escape	エスケープ	0xff1b
KEY_Left	左カーソル	0xff51
KEY_Up	上カーソル	0xff52
KEY_Right	右カーソル	0xff53
KEY_Down	下カーソル	0xff54
KEY_Page_Up	ページアップ	0xff55
KEY_Page_Down	ページダウン	0xff56
KEY_Insert	挿入	0xff63
KEY_Delete	削除	0xffff
KEY_Home	ホーム	0xff50
KEY_End	エンド	0xff57
KEY_F1	F1	0xffbe
KEY_F2	F2	0xffbf
KEY_F3	F3	0xffc0
KEY_F4	F4	0xffc1
KEY_F5	F5	0xffc2
KEY_F6	F6	0xffc3
KEY_F7	F7	0xffc4
KEY_F8	F8	0xffc5
KEY_F9	F9	0xffc6
KEY_F10	F10	0xffc7
KEY_F11	F11	0xffc8
KEY_F12	F12	0xffc9
KEY_Shift_L	シフト(左)	0xffe1
KEY_Shift_R	シフト(右)	0xffe2
KEY_Control_L	コントロール(左)	0xffe3
KEY_Control_R	コントロール(右)	0xffe4
KEY_Meta_L	メタ(左)	0xffe7
KEY_Meta_R	メタ(右)	0xffe8
KEY_Alt_L	Alt(左)	0xffe9
KEY_Alt_R	Alt(右)	0xffea
KEY_0	0	0x030
KEY_1	1	0x031
KEY_2	2	0x032
KEY_3	3	0x033
KEY_4	4	0x034
KEY_5	5	0x035

KEY_6	6	0x036
KEY_7	7	0x037
KEY_8	8	0x038
KEY_9	9	0x039
KEY_A	A	0x041
KEY_B	B	0x042
KEY_C	C	0x043
KEY_D	D	0x044
KEY_E	E	0x045
KEY_F	F	0x046
KEY_G	G	0x047
KEY_H	H	0x048
KEY_I	I	0x049
KEY_J	J	0x04a
KEY_K	K	0x04b
KEY_L	L	0x04c
KEY_M	M	0x04d
KEY_N	N	0x04e
KEY_O	O	0x04f
KEY_P	P	0x050
KEY_Q	Q	0x051
KEY_R	R	0x052
KEY_S	S	0x053
KEY_T	T	0x054
KEY_U	U	0x055
KEY_V	V	0x056
KEY_W	W	0x057
KEY_X	X	0x058
KEY_Y	Y	0x059
KEY_Z	Z	0x05a
KEY_a	a	0x061
KEY_b	b	0x062
KEY_c	c	0x063
KEY_d	d	0x064
KEY_e	e	0x065
KEY_f	f	0x066
KEY_g	g	0x067
KEY_h	h	0x068
KEY_i	i	0x069
KEY_j	j	0x06a
KEY_k	k	0x06b
KEY_l	l	0x06c
KEY_m	m	0x06d
KEY_n	n	0x06e
KEY_o	o	0x06f
KEY_p	p	0x070
KEY_q	q	0x071
KEY_r	r	0x072
KEY_s	s	0x073
KEY_t	t	0x074
KEY_u	u	0x075
KEY_v	v	0x076
KEY_w	w	0x077
KEY_x	x	0x078
KEY_y	y	0x079
KEY_z	z	0x07a

(3)Color names

Name	Description	Notes
:ALICE_BLUE		#F0F8FFFF
:ALIZARIN_CRIMSON		#E32637FF
:AMARANTH		#E62B50FF
:AMBER		#FFBF00FF
:AMETHYST		#9966CCFF
:ANTIQUE_WHITE		#FAEBD8FF
:APRICOT		#FBCEB2FF
:AQUA		#00FFFFFF
:AQUAMARINE		#80FFD5FF
:ASPARAGUS		#7CA15BFF
:AZURE		#0080FFFF
:BEIGE		#F5F5DDFF
:BISQUE		#FFE5C4FF
:BISTRE		#3E2B1FFF
:BLACK		#000000FF
:BLANCHED_ALMOND		#FFEBCDFF
:BLAZE_ORANGE		#FF6600FF
:BLUE		#0000FFFF
:BLUE_VIOLET		#8A2BE3FF
:BONDI_BLUE		#0095B6FF
:BRIGHT_GREEN		#66FF00FF
:BRIGHT_TURQUOISE		#08E8DFFF
:BROWN		#964C00FF
:BUFF		#F0DD82FF
:BURGUNDY		#800020FF
:BURLY_WOOD		#DFB887FF
:BURNT_ORANGE		#CC5500FF
:BURNT_SIENNA		#E97551FF
:BURNT_UMBER		#8A3324FF
:CADET_BLUE		#5F9FA1FF
:CAMOUFLAGE_GREEN		#79866CFF
:CARDINAL		#C41E3BFF
:CARMINE		#960018FF
:CARNATION		#F95A61FF
:CARROT_ORANGE		#ED9121FF
:CELADON		#ADE2B0FF
:CERISE		#DF3163FF
:CERULEAN		#007CA8FF
:CERULEAN_BLUE		#2A52BEFF
:CHARTREUSE		#80FF00FF
:CHARTREUSE_YELLOW		#E0FF00FF
:CHESTNUT		#CD5C5CFF
:CHOCOLATE		#D36A1EFF
:CINNAMON		#7C4000FF
:COBALT		#0048ACFF
:COPPER		#B87433FF
:COPPER_ROSE		#996666FF
:CORAL		#FF8050FF
:CORAL_RED		#FF4141FF
:CORN		#FBEC5DFF
:CORNFLOWER_BLUE		#6495EDFF
:CORNSILK		#FFF8DDFF

:CREAM		#FFFDD1FF
:CRIMSON		#DD143DFF
:CYAN		#00FFFFFF
:DARK_BLUE		#00008BFF
:DARK_CYAN		#008B8BFF
:DARK_GOLDENROD		#B8860CFF
:DARK_GRAY		#AAAAAAFF
:DARK_GREEN		#006400FF
:DARK_KHAKI		#BDB76CFF
:DARK_MAGENTA		#8B008BFF
:DARK_OLIVE_GREEN		#556C2FFF
:DARK_ORANGE		#FF8C00FF
:DARK_ORCHID		#9932CCFF
:DARK_POWDER_BLUE		#003399FF
:DARK_RED		#8B0000FF
:DARK_SALMON		#E9967BFF
:DARK_SEA_GREEN		#8FBC8FFF
:DARK_SLATE_BLUE		#493E8BFF
:DARK_SLATE_GRAY		#2F4F4FFF
:DARK_TURQUOISE		#00CED2FF
:DARK_VIOLET		#9400D4FF
:DEEP_PINK		#FF1493FF
:DEEP_SKY_BLUE		#00BFFFFF
:DENIM		#1660BDFF
:DIM_GRAY		#6A6A6AFF
:DODGER_BLUE		#1E90FFFF
:EGGPLANT		#990066FF
:EMERALD		#50C879FF
:FALU_RED		#801818FF
:FERN_GREEN		#4F7A43FF
:FIRE_BRICK		#B22222FF
:FLAX		#EEDD82FF
:FLORAL_WHITE		#FFFAF0FF
:FOREST_GREEN		#228B22FF
:FRENCH_ROSE		#F64B8AFF
:FUCHSIA		#FF00FFFF
:GAINSBORO		#DDDDDDFF
:GAMBOGE		#E59C0FFF
:GHOST_WHITE		#F8F8FFFF
:GOLD		#FFD800FF
:GOLDENROD		#DBA620FF
:GRAY		#808080FF
:GRAY_ASPARAGUS		#475946FF
:GREEN		#00FF00FF
:GREEN_YELLOW		#AEFF2FFF
:HARLEQUIN		#40FF00FF
:HELIOTROPE		#E074FFFF
:HOLLYWOOD_CERISE		#F400A2FF
:HONEYDEW		#F0FFF0FF
:HOT_MAGENTA		#FF00CCFF
:HOT_PINK		#FF6AB5FF
:INDIAN_RED		#CD5C5CFF
:INDIGO		#4C0082FF
:INTERNATIONAL_KLEIN_BLUE		#002FA8FF
:INTERNATIONAL_ORANGE		#FF4F00FF
:IVORY		#FFFFF0FF

:JADE		#00A96CFF
:KHAKI		#C3B191FF
:KHAKI_X11		#F0E78CFF
:LAVENDER		#B57FDDFF
:LAVENDER_BLUE		#CCCCFFFF
:LAVENDER_BLUSH		#FFF0F5FF
:LAVENDER_GRAY		#BDBBD8FF
:LAVENDER_PINK		#FBAFD3FF
:LAVENDER_ROSE		#FBA1E3FF
:LAWN_GREEN		#7DFC00FF
:LEMON		#FDE911FF
:LEMON_CHIFFON		#FFFACDFF
:LIGHT_BLUE		#AED9E7FF
:LIGHT_CORAL		#F08080FF
:LIGHT_CYAN		#E1FFFFFF
:LIGHT_GOLDENROD_YELLOW		#FAFAD3FF
:LIGHT_GREEN		#90EE90FF
:LIGHT_GREY		#D4D4D4FF
:LIGHT_PINK		#FFB6C1FF
:LIGHT_SALMON		#FFA17BFF
:LIGHT_SEA_GREEN		#20B2ABFF
:LIGHT_SKY_BLUE		#87CEFAFF
:LIGHT_SLATE_GRAY		#788899FF
:LIGHT_STEEL_BLUE		#B1C4DFFF
:LIGHT_YELLOW		#FFFFE1FF
:LILAC		#C8A3C8FF
:LIME		#BFFF00FF
:LIME_GREEN		#32CD32FF
:LINEN		#FAF0E7FF
:MAGENTA		#FF00FFFF
:MALACHITE		#0CDB51FF
:MAROON		#800000FF
:MAUVE		#E1B1FFFF
:MEDIUM_AQUAMARINE		#66CDABFF
:MEDIUM_BLUE		#0000CDFF
:MEDIUM_CARMINE		#B04136FF
:MEDIUM_LAVENDER		#EE82EEFF
:MEDIUM_ORCHID		#BA55D4FF
:MEDIUM_PURPLE		#9371DCFF
:MEDIUM_SEA_GREEN		#3DB372FF
:MEDIUM_SLATE_BLUE		#7C69EEFF
:MEDIUM_SPRING_GREEN		#00FA9BFF
:MEDIUM_TURQUOISE		#49D2CCFF
:MEDIUM_VIOLET_RED		#C71685FF
:MIDNIGHT_BLUE		#003366FF
:MINT_CREAM		#F5FFFAFF
:MINT_GREEN		#98FF98FF
:MISTY_ROSE		#FFE5E2FF
:MOCCASIN		#FFE5B5FF
:MOSS_GREEN		#AEE0AEFF
:MOUNTBATTEN_PINK		#997B8DFF
:MUSTARD		#FFDC58FF
:NAVAJO_WHITE		#FFDFAEFF
:NAVY_BLUE		#000080FF
:OCHRE		#CC7822FF
:OLD_GOLD		#D0B53CFF

:OLD_LACE		#FDF5E7FF
:OLD_LAVENDER		#7A6979FF
:OLD_ROSE		#C08081FF
:OLIVE		#808000FF
:OLIVE_DRAB		#6C8E23FF
:ORANGE		#FF8000FF
:ORANGE_COLOR_WHEEL		#FF8000FF
:ORANGE_PEEL		#FFA100FF
:ORANGE_RED		#FF4600FF
:ORANGE_WEB		#FFA600FF
:ORCHID		#DB71D7FF
:PALE_GOLDENROD		#EEE8ABFF
:PALE_GREEN		#98FB98FF
:PALE_TURQUOISE		#B0EEEEFF
:PALE_VIOLET_RED		#DC7193FF
:PAPAYA_WHIP		#FFEFD6FF
:PASTEL_GREEN		#78DE78FF
:PASTEL_PINK		#FFD2DDFF
:PEACH		#FFE6B5FF
:PEACH_ORANGE		#FFCC99FF
:PEACH_PUFF		#FFDBB9FF
:PEACH_YELLOW		#FAE0AEFF
:PEAR		#D2E331FF
:PERIWINKLE		#CCCCFFFF
:PERSIAN_BLUE		#1C3ABBFF
:PERSIAN_GREEN		#00A793FF
:PERSIAN_INDIGO		#32127BFF
:PERSIAN_PINK		#F780BEFF
:PERSIAN_RED		#CC3333FF
:PERSIAN_ROSE		#FF1CB2FF
:PERU		#CD8540FF
:PINE_GREEN		#017A70FF
:PINK		#FFC0CBFF
:PINK_ORANGE		#FF9966FF
:PLUM		#DEA1DEFF
:POMEGRANATE		#F34823FF
:POWDER_BLUE		#B1E1E7FF
:POWDER_BLUE_WEB		#B1E1E7FF
:PRUSSIAN_BLUE		#003153FF
:PUCE		#CC8899FF
:PUMPKIN		#FF7618FF
:PURPLE		#660099FF
:RAW_UMBER		#744B12FF
:RED		#FF0000FF
:RED_VIOLET		#C71685FF
:ROBIN_EGG_BLUE		#00CCCCFF
:ROSE		#FF0080FF
:ROSY_BROWN		#BC8F8FFF
:ROYAL_BLUE		#426AE2FF
:RUSSET		#80471BFF
:RUST		#B7420EFF
:SADDLE_BROWN		#8B4613FF
:SAFETY_ORANGE		#FF6600FF
:SAFFRON		#F4C430FF
:SALMON		#FF8C6AFF
:SANDY_BROWN		#F4A560FF

:SANGRIA		#92000AFF
:SAPPHIRE		#082567FF
:SCARLET		#FF2400FF
:SCHOOL_BUS_YELLOW		#FFD900FF
:SEA_GREEN		#2E8B57FF
:SEASHELL		#FFF5EEFF
:SELECTIVE_YELLOW		#FFBA00FF
:SEPIA		#714314FF
:SHOCKING_PINK		#FC0FC0FF
:SIENNA		#A1522DFF
:SILVER		#C0C0C0FF
:SKY_BLUE		#87CEEBFF
:SLATE_BLUE		#6B5ACDFF
:SLATE_GRAY		#718090FF
:SMALT		#003399FF
:SNOW		#FFFAFAFF
:SPRING_GREEN		#00FF80FF
:STEEL_BLUE		#4782B5FF
:SWAMP_GREEN		#ADB78EFF
:TAN		#D3B58CFF
:TANGERINE		#FFCC00FF
:TAUPE		#493D32FF
:TAWNY		#CD5700FF
:TEA_GREEN		#D1F0C0FF
:TEAL		#008080FF
:TENNE		#CD5700FF
:TERRA_COTTA		#E3735BFF
:THISTLE		#D9BFD9FF
:TOMATO		#FF6348FF
:TURQUOISE		#30D6C8FF
:ULTRAMARINE		#120A8FFF
:VERMILION		#FF4D00FF
:VIOLET		#8B00FFFF
:VIOLET_EGGPLANT		#991299FF
:VIRIDIAN		#41826EFF
:WHEAT		#F5DFB3FF
:WHITE		#FFFFFFFF
:WHITE_SMOKE		#F5F5F5FF
:WISTERIA		#C9A1DDFF
:YELLOW		#FFFF00FF
:YELLOW_GREEN		#9BCD32FF
:ZINNWALDITE		#EBC2B0FF