Documentation de Nano Lua 2.3

Traduction par Reylak et mise à jour par Quent42340

Quelques variables utiles

MICROLUA_VERSION : *version de Micro Lua (sous forme d'un String)*
SCREEN_WIDTH : *largeur des écrans*
SCREEN_HEIGHT : *hauteur des écrans*
NB_FPS : *nombre de frames (images) par seconde, automatiquement mis à jour chaque seconde. Vous pouvez l'afficher grâce à un screen.print(). Ne pas afficher dans la console de déboguage.*
SCREEN_UP, SCREEN_DOWN : *numéro de l'écran sur lequel afficher les éléments. Voir les exemples pour de plus amples détails.*
RAM, VRAM : *destination du chargement de certains objets. Voir les exemples pour de plus amples détails.*
ATTR_X1, ATTR_Y1, ATTR_X2, ATTR_Y1, ATTR_X3, ATTR_Y3, ATTR_COLOR, ATTR_COLOR1, ATTR_COLOR2, ATTR_COLOR3, ATTR_COLOR4, ATTR_TEXT, ATTR_VISIBLE, ATTR_FONT, ATTR_IMAGE : *attributs utilisés par les canevas.*

L'écran

(Toutes les méthodes sont partagées)

Void render()
Actualise l'écran.

Void startDrawing() & stopDrawing() OBSOLETE DEPUIS LA VERSION 3.0
Toutes vos opérations graphiques doivent être placées entre ces deux fonctions.

Void screen.switch()
Intervertit l'affichage des écrans.

Void screen.print(ecran, x, y, texte [, couleur])
Affiche un texte à l'écran.
ecran (Number) : écran sur lequel dessiner (SCREEN_UP ou SCREEN_DOWN)
x (Number) : abscisse du coin haut-gauche du texte
y (Number) : ordonnée du coin haut-gauche du texte
texte (String) : texte à afficher
couleur (Color) : couleur du texte

Void screen.printFont(ecran, x, y, texte, couleur, police)
Affiche un texte à l'écran dans une police spéciale.
ecran (Number) : écran sur lequel dessiner (SCREEN_UP ou SCREEN_DOWN)
x (Number) : abscisse du coin haut-gauche du texte
y (Number) : ordonnée du coin haut-gauche du texte
texte (String) : texte à afficher
couleur (Color) : couleur du texte
police (Font) : police spéciale

Void screen.blit(ecran, x, y, image [,sourceX, sourceY] [, largeur, hauteur])
Blitte (affiche) une image à l'écran.
ecran (Number) : écran sur lequel afficher l'image (SCREEN_UP ou SCREEN_DOWN)
x (Number) : abscisse du coin haut-gauche de l'image
y (Number) : ordonnée du coin haut-gauche de l'image
image (Image) : image à blitter
sourceX, sourceY (Number) : coordonnées de la partie de l'image source à blitter
largeur, hauteur (Number) : dimensions de la partie de l'image source à blitter

Void screen.drawLine(ecran, x0, y0, x1, y1, couleur)
Dessine une ligne à l'écran.
ecran (Number) : écran sur lequel dessiner (SCREEN_UP ou SCREEN_DOWN)
x0, y0, x1, y1 (Number) : coordonnées de la ligne
couleur (Color) : couleur de la ligne

Void screen.drawRect(ecran, x0, y0, x1, y1, couleur)
Dessine un rectangle à l'écran.
ecran (Number) : écran sur lequel dessiner (SCREEN_UP ou SCREEN_DOWN)
x0, y0, x1, y1 (Number) : coordonnées du rectangle
couleur (Color) : couleur du rectangle

Void screen.drawFillRect(ecran, x0, y0, x1, y1, couleur)
Dessine un rectangle plein à l'écran.
ecran (Number) : écran sur lequel dessiner (SCREEN_UP ou SCREEN_DOWN)
x0, y0, x1, y1 (Number) : coordonnées du rectangle plein
couleur (Color) : couleur du rectangle plein

Void screen.drawGradientRect(ecran, x0, y0, x1, y1, couleur1, couleur2, couleur3, couleur4)
Dessine un rectangle dégradé à l'écran.
ecran (Number) : écran sur lequel dessiner (SCREEN_UP ou SCREEN_DOWN)
x0, y0, x1, y1 (Number) : coordonnées du rectangle dégradé
couleur1, couleur2, couleur3, couleur4 (Color) : couleurs du rectangle dégradé

Void screen.drawTextBox(ecran, x0, y0, x1, y1, texte [, couleur])
Affiche une zone de texte à l'écran.
ecran (Number) : écran sur lequel dessiner (SCREEN_UP ou SCREEN_DOWN)
x0, y0, x1, y1 (Number) : coordonnées de la zone de texte
texte (String) : texte à afficher
couleur (Color) : couleur de la zone de texte

Couleurs

Color Color.new(r, v, b)
Crée une nouvelle couleur.
r (Number) : masque rouge (de 0 à 31 inclus)
v (Number) : masque vert (de 0 à 31 inclus)
b (Number) : masque bleu (de 0 à 31 inclus)

Contrôles

Void Controls.read()
Met à jour les commandes.

Stylet :

Stylus.X : abscisse du stylet
Stylus.Y : ordonnée du stylet
Stylus.held : état appuyé sur l'écran du stylet
Stylus.released : état relâché du stylet
Stylus.doubleClick : vaut true si le stylet effectue un double-clic
Stylus.deltaX : delta X (accroissement de X) du mouvement du stylet
Stylus.deltaY : delta Y (accroissement de Y) du mouvement du stylet
Stylus.newPress : vaut true si le stylet effectue une nouvelle pression

Boutons :

([KEY] peut prendre une des valeurs suivantes : A, B, X, Y, R, L, Start, Select, Up, Down, Left, Right)

Keys.held.[KEY] : état enfoncé du bouton
Keys.released.[KEY] : état relâché du bouton
Keys.newPress.[KEY] : vaut true si le bouton effectue une nouvelle pression

Images

Image Image.load(chemin, destination)
Crée une nouvelle image depuis un fichier image (PNG, JPG – JPEG ou GIF).
chemin (String) : chemin d'accès du fichier image à charger
destination (Number) : destination de l'image dans la mémoire (RAM ou VRAM)

Void Image.destroy(image)
Détruit une image.
Pour détruire complètement l'image, faire :
Image.destroy(monImage)
monImage = nil
image (Image) : image à détruire

Number Image.width(image)
Obtient la longueur d'une image.
image (Image) : image à utiliser

Number Image.height(image)
Obtient la largeur d'une image.
image (Image) : image à utiliser

Void Image.scale(image, longueur, largeur)
Agrandit (ou réduit) l'image.
image (Image) : image à modifier
longueur (Number) : nouvelle longueur de l'image
largeur (Number) : nouvelle largeur de l'image

Void Image.rotate(image, angle [, xCentre, yCentre])
Fait pivoter l'image autour du centre de rotation indiqué.
image (Image) : image à modifier
angle (Number) : angle de rotation (entre 0 et 511)
xCentre (Number) : abscisse du centre de rotation
yCentre (Number) : ordonnée du centre de rotation

Void Image.rotateDegree(image, angle [, xCentre, yCentre])
Fait pivoter l'image autour du centre de rotation indiqué. L'angle est en degrés.
image (Image) : image à modifier
angle (Number) : angle de rotation, en degrés (entre 0 et 360)
xCentre (Number) : abscisse du centre de rotation
yCentre (Number) : ordonnée du centre de rotation

Void Image.mirrorH(image)
Réalise une symétrie horizontale de l'image.
image (Image) : image à modifier

Void Image.mirrorV(image)
Réalise une symétrie verticale de l'image.
image (Image) : image à modifier

Void Image.setTint(image, couleur)
Définit la teinte de l'image.
image (Image) : image à modifier
couleur (Color) : couleur de l'image

Timers (chronomètres) en millisecondes

Timer Timer.new()
Crée un nouveau timer ; vous pouvez le démarrer.

Number timer:time()
Renvoie le temps d'un timer.

Void Timer:start()
Démarre un timer.

Void Timer:stop()
Arrête un timer.

Void timer:reset()
Réinitialise un timer.

Sprites (succession d'images – les frames – pour créer une animation)

Sprite Sprite.new(chemin, longueurFrame, largeurFrame, destination)
Crée un sprite depuis un fichier image.
chemin (String) : chemin d'accès du fichier qui contient le sprite
longueurImage (Number) : longueur d'une frame
largeurImage (Number) : largeur d'une frame
destination (Number) : destination en mémoire (RAM ou VRAM)

Void sprite:drawFrame(ecran, x, y, numFrame)
Affiche une frame du sprite.
ecran (Nombre) : écran sur lequel afficher (SCREEN_UP ou SCREEN_DOWN)
x (Nombre) : abscisse du coin haut-gauche de la frame
y (Nombre) : ordonnée du coin haut-gauche de la frame
numFrame (Number) : numéro de la frame à afficher

Void sprite:addAnimation(tabAnim, delais)
Crée une animation.
tabAnim (Table) : table des frames de l'animation
delais (Number) : délais entre chaque frame

Void sprite:playAnimation(ecran, x, y, numAnim)
Joue une animation à l'écran.
ecran (Number) : écran sur lequel afficher (SCREEN_UP ou SCREEN_DOWN)
x (Number) : abscisse du coin haut-gauche de l'animation
y (Number) : ordonnée du coin haut-gauche de l'animation
numAnim (Number) : numéro de l'animation à jouer

Void sprite:resetAnimation(numAnim)
Réinitialise une animation.
numAnim (Number) : numéro de l'animation à réinitialiser.

Void sprite:startAnimation(numAnim)
Démarre une animation.
numAnim (Number) : numéro de l'animation à démarrer

Void sprite:stopAnimation(numAnimation)
Arrête une animation.
numAnim (Number) : numéro de l'animation à arrêter

Boolean sprite:isAnimationAtEnd(numAnim)
Renvoie true si l'animation a affiché la dernière frame.
numAnim (Number) : numéro de l'animation

Déboguage

(Toutes les méthodes sont partagées)

Void Debug.ON()
Active le mode de déboguage.

Void Debug.OFF()
Désactive le mode déboguage.

Void Debug.print(texte)
Affiche une ligne de déboguage.
texte (String) : texte à afficher
Void Debug.clear()
Efface la console de déboguage.

Void Debug.setColor(couleur)
Définit la couleur du texte de déboguage.
couleur (Color) : couleur du texte

Système

(Toutes les méthodes sont partagées)

String System.currentDirectory()
Obtient le répertoire actuel.

Void System.changeDirectory(chemin)
Change le répertoire actuel.
chemin (String) : chemin du dossier

Void System.remove(nom)
Supprime un fichier ou un répertoire vide.
nom (String) : nom du fichier ou du dossier à supprimer

Void System.rename(ancienNom, nouveauNom)
Renomme un fichier ou un répertoire vide.
ancienNom (String) : nom du fichier ou du dossier à renommer
nouveauNom (String) : nouveau nom du fichier ou du dossier

Void System.makeDirectory(nom)
Crée un nouveau répertoire.
nom (String) : chemin d'accès et nom du dossier

Table System.listDirectory(chemin)
Liste tous les fichiers et répertoires d'un dossier.
chemin (String) : chemin d'accès du dossier à lister

(voir les exemples pour plus de détails)

DSUser

*(Ceci sont des variables **et** des fonctions)*

String DSUser.UserName
Obtient le pseudo de l'utilisateur.

String DSUser.UserMessage
Obtient le message de l'utilisateur.

Number DSUser.UserLangNum
Obtient le numéro de la langue de la DS (O = Japonais, 1 = Anglais, 2 = Français, 3 = Allemand, 4 = Italien et 5 = Espagnol).

String DSUser.UserLangName
Obtient le nom de la langue (Japanese, English, French, German, Italian et Spanish).

Number DSUser.UserColorNum
Obtient le numéro de la couleur de skin de la DS (0 = Gris, 1 = Marron, 2 = Rouge, 3 = Rose, 4 = Orange, 5 = Jaune, 6 = Lime, 7 = Vert, 8 = Vert foncé, 9 = Teal, 10 = Bleu clair, 11 = Bleu, 12 = Bleu foncé, 13 = Violet foncé, 14 = Violet clair, 15 = Rose foncé).

String DSUser.UserColorName
Obtient le nom de la couleur de skin de la DS (Gray, Maroon, Red, Pink, Orange, Yellow, Lime, Green, Dark Green, Teal, Light Blue, Blue, Dark Blue, Dark Purple, Light Purple, Dark Pink).

Number DSUser.BirthMonth
Obtient le mois d'anniversaire de l'utilisateur.

Number DSUser.BirthDay
Obtient le jour d'anniversaire de l'utilisateur.

Number DSUser.AlarmHour
Obtient l'heure de l'alarme de la DS.

Number DSUser.AlarmMin
Obtient la minute de l'alarme de la DS.

Number DSUser.LidClosed
Retourne true si la lid est fermée.

Void DSUser.SetScreenLight(screen, light)
Définit si l'écran sélectionné est allumé ou éteint .
ecran (Number) : écran qui sera éteint ou allumé (SCREEN_UP ou SCREEN_DOWN)
light (Number) : Définit si l'écran sélectionné est allumé (LIGHT_ON) ou éteint (LIGHT_OFF)

Void DSUser.SetDSLBrightness(brightness)
Définit la luminosité des écrans (Uniquement pour DS Lite).
brightness (Number) : Définit la luminosité des écrans (0, 1, 2 ou 3, 0 étant le plus obscur et 3 étant le plus lumineux)

Void DSUser.SetLedBlink(blink, speed)
Définit le clignotement de la LED.
blink (Number) : Définit si la LED clignote (START_BLINK) ou si elle ne clignote pas (STOP_BLINK)
speed (Number) : Définit si la LED clignote vite, comme pour une communication WI-FI, (FAST_BLINK) ou lentement (SLOW_BLINK)

Polices

Font Font.load(chemin)
Crée une nouvelle police depuis un fichier de police (au format oslib et µLibrairy).
chemin (String) : chemin d'accès du fichier police à charger

Number Font.getCharHeight(police)
Obtient la hauteur des caractères d'une police.
police (Font) : police à utiliser

Number Font.getStringWidth(police, texte)
Obtient la largeur d'un texte avec une police spéciale.
police (Font) : police à utiliser
texte (String) : texte

Cartes

Map Map.new(image, fichierCarte, longueur, largeur, longueurTuile, largeurTuile)
Crée une nouvelle carte à partir d'un fichier carte.
image (Image) : image qui contient les tuiles
fichierCarte (String) : chemin d'accès au fichier carte (.map)
longueur (Number) : longueur de la carte en tuile
largeur (Number) : largeur de la carte en tuile
longueurTuile (Number) : longueur d'une tuile en pixel
largeurTuile (Number) : largeur d'une tuile en pixel

Void Map.destroy(carte)
Détruit une carte.
carte (Map) : carte à détruire

Void Map.draw(ecran, carte, x, y, longueur, largeur)
Affiche une carte.
ecran (Number) : écran sur lequel afficher la carte (peut être SCREEN_UP ou SCREEN_DOWN)
carte (Map) : carte à afficher
x (Number) : abscisse du coin haut-gauche de la carte
y (Number) : ordonnée du coin haut-gauche de la carte
longueur (Number) : nombre de tuiles à afficher sur la longueur
largeur (Number) : nombre de tuiles à afficher sur la largeur

Void Map.scroll(carte, x, y)
Fait défiler une carte.
carte (Map) : carte à faire défiler
x (Number) : nombre de tuiles à faire défiler sur la longueur
y (Number) : nombre de tuiles à faire défiler sur la largeur

Void Map.space(carte, x, y)

Définit l'espace entre chaque tuile de la carte.
carte (Map) : carte à modifier
x (Number) : espace sur la longueur entre chaque tuile
y (Number) : espace sur la largeur entre chaque tuile

Void Map.setTile(carte, x, y, tuile)
Change la valeur d'une tuile.
carte (Map) : carte à modifier
x (Number) : numéro de la colonne de la tuile à changer dans la carte
y (Number) : numéro de la ligne de la tuile à changer dans la carte
tuile (Number) : nouvelle valeur de tuile

Number Map.getTile(carte, x ,y)
Obtient une valeur de tuile.
carte (Map) : carte qui contient la tuile voulue
x (Number) : numéro de la colonne de la tuile à obtenir
y (Number) : numéro de la ligne de la tuile à obtenir

Cartes défilantes

ScrollMap ScrollMap.new(image, fichierCarte, longueur, largeur, longueurTuile, largeurTuile)
Crée une nouvelle carte défilante.
image (Image) : image qui contient les tuiles
fichierCarte (String) : chemin d'accès du fichier carte (.map)
longueur (Number) : longueur de la carte en tuile
largeur (Number) : largeur de la carte en tuile
longueurTuile (Number) : longueur d'une tuile en pixel
largeurTuile (Number) : largeur d'une tuile en pixel

Void ScrollMap.destroy(carteDefilante)
Détruit une carte défilante.
carteDéfilante (ScrollMap) : carte défilante à détruire

Void ScrollMap.draw(carteDefilante)
Affiche une carte défilante.
carteDéfilante (ScrollMap) : carte défilante à afficher

Void ScrollMap.scroll(carteDefilante, x, y)
Fait défiler une carte défilante.
carteDefilante (ScrollMap) : carte défilante à faire défiler
x (Number) : défilement horizontal en pixel
y (Number) : défilement horizontal en pixel

Canevas

Canvas Canvas.new()
Crée un nouveau canevas.
Void Canvas.destroy(canevas)

Détruit un canevas. Doit être suivi de canevas = nil.
canevas (Canvas) : canevas à détruire

CanvasObject Canvas.newLine(x1, y1, x2, y2, couleur)
Crée une nouvelle ligne.
x1, y1, x2, y2 (Number) : coordonnées de la ligne
couleur (Color) : couleur de la ligne

CanvasObject Canvas.newPoint(x1, y1, couleur)
Crée un nouveau point.
x1, y1 (Number) : coordonnées du point
couleur (Color) : couleur du point

CanvasObject Canvas.newRect(x1, y1, x2, y2, couleur)
Crée un nouveau rectangle.
x1, y1, x2, y2 (Number) : coordonnées du rectangle
couleur (Color) : couleur du rectangle

CanvasObject Canvas.newFillRect(x1, y1, x2, y2, couleur)
Crée un nouveau rectangle plein.
x1, y1, x2, y2 (Number) : coordonnées du rectangle
couleur (Color) : couleur du rectangle

CanvasObject Canvas.newGradientRect(x1, y1, x2, y2, couleur1, couleur2, couleur3, couleur4)
Crée un nouveau rectangle dégradé.
x1, y1, x2, y2 (Number) : coordonnées du rectangle
couleur1, couleur2, couleur3, couleur4 (Color) : couleurs du rectangle

CanvasObject Canvas.newText(x, y, texte [, couleur])
Crée un nouveau texte.
x (Nombre) : abscisse du coin haut-gauche du texte
y (Nombre) : ordonnée du coin haut-gauche du texte
texte (String) : texte
couleur (Color) : couleur du texte

CanvasObject Canvas.newTextFont(x, y, texte, couleur, police)
Crée un nouveau texte avec une police spéciale.
x (Nombre) : abscisse du coin haut-gauche du texte
y (Nombre) : ordonnée du coin haut-gauche du texte
texte (String) : texte
couleur (Color) : couleur du texte
police (Font) : police spéciale du texte

CanvasObject Canvas.newTextBox(x1, y1, x2 , y2, texte [, couleur])
Crée une nouvelle zone de texte.
x1, y1, x1, y2 (Number) : coordonnées de la zone de texte
texte (String) : texte de la zone de texte
couleur (Color) : couleur de la zone de texte

CanvasObject Canvas.newImage(x1, y1, image [, x2, y2][, x3, y3])
Crée une nouvelle image.
x1, y1 (Number) : coordonnées du coin haut-gauche de l'image
image (Image) : image
x2, y2 (Number) : coordonnées du coin haut-gauche du rectangle à prendre dans l'image source
x3, y3 (Number) : longueur et largeur du rectangle à prendre dans l'image source

Void Canvas.add(canevas, objet)
Ajoute un CanvasObject dans un canevas.

canevas (Canvas) : canevas à modifier
objet (CanvasObject) : objet à ajouter

Void Canvas.draw(ecran, canevas, x, y,)
Affiche un canevas à l'écran.
ecran (Number) : écran sur lequel dessiner (SCREEN_UP ou SCREEN_DOWN)
canevas (Canvas) : canevas à afficher
x (Number) : abscisse du coin haut-gauche de l'image
y (Number) : ordonnée du coin haut-gauche de l'image

Void Canvas.setAttr(objet, nomAttr, valeurAttr)
Définit la valeur d'un attribut d'un CanvasObject.
objet (CanvasObject) : objet à modifier
nomAttr (Constant) : attribut à modifier. Doit être soir la forme ATTR_XXX. Voir « Quelques variables utiles » pour de plus amples informations.
valeurAttr (?) : nouvelle valeur de l'attribut. Le type doit être le même que celui de l'attribut.

? Canvas.getAttr(objet, nomAttr)
Obtient la valeur d'un attribut. Le type de retour dépend de l'attribut demandé.
objet (CanvasObject) : objet à utiliser
nomAttr (Constant) : attribut dont la valeur est demandée. Doit être sous la forme ATTR_XXX. Voir « Quelques variables utiles » pour de plus amples informations

Rumble

Boolean Rumble.isInserted()
Vérifie qu'un pack rumble (vibration) soit inséré.

Void Rumble.set(statut)
Définit le statut du rumble.
statut (Boolean) : statut du rumble (true : marche ; false : arrêt)

Motion (détection de mouvement)

Boolean Motion.init()
Initialise le système Motion si un périphérique Motion est détecté. Renvoie true si un périphérique Motion est détecté.

Void Motion.calibrate()
Effectue le calibrage du système Motion.

Number Motion.readX()
Renvoie l'inclinaison X du système Motion.

Number Motion.readY()
Renvoie l'inclinaison Y du système Motion.

Number Motion.readZ()
Renvoie l'inclinaison Z du système Motion.

Number Motion.accelerationX()
Retourne l'accélération X du système Motion.

Number Motion.accelerationY()
Retourne l'accélération Y du système Motion.

Number Motion.accelerationZ()
Retourne l'accélération Z du système Motion.

Number Motion.readGyro()
Retourne la valeur de gyro du système Motion.

Number Motion.rotation()
Retourne la valeur de rotation du système Motion.

Date et heure

DateTime DateTime.new()
Crée un nouvel objet DateTime.

DateTime DateTime.getCurrentTime()
Crée un nouvel objet DateTime avec l'heure et la date courantes.

Attributs :
year (an), month (mois), day (jour), hour (heure), minute, second (seconde)

(voir les exemples pour plus de détails)

Wifi

Void Wifi.connectWFC()
Connecte la DS à la connexion Wifi. Utilise la configuration du firmware. Par conséquent, vous devez configurer votre connexion avec un jeu DS officiel.

Void Wifi.disconnect()
Déconnecte la DS de la connexion Wifi.

Socket Wifi.createTCPSocket(hote, port)
Crée un socket TCP vers un serveur.
hote (String) : nom de domaine ou adresse IP du serveur
port (Number) : port à utiliser

Socket Wifi.createUDPSocket(hote, port)
Crée un socket UDP vers un serveur.
hote (String) : nom de domaine ou adresse IP du serveur
port (Number) : port à utiliser

Void Wifi.closeSocket(socket)
Ferme un socket (TCP comme UDP).

socket (Socket) : socket à fermer

Void Wifi.send(socket, donnees)
Envoie des données à un serveur en passant par un socket.
socket (Socket) : socket à utiliser
donnees (String) : données à envoyer

String Wifi.receive(socket, longueur)
Reçoit des données depuis un serveur en passant par un socket.
socket (Socket) : socket à utiliser
longueur (Number) : taille des données à recevoir

Le son

Banques de son :

Void Sound.loadBank(nomFichier)
Charge une banque de son depuis un fichier en mémoire.
nomFichier (String) : chemin du fichier à charger

Void Sound.unloadBank()
Décharge la banque de son de la mémoire.

Mods :

Void Sound.loadMod(index)
Charge un module en mémoire.
index (Number) : index du module à charger

Void Sound.unloadMod(index)
Décharge un module de la mémoire.
index (Number) : index du module à décharger

Void Sound.startMod(index, modeLecture)
Lance la lecture d'un module déjà chargé en mémoire.
index (Number) : index du module à jouer
modeLecture (Number) : mode de lecture (peut être PLAY_ONCE – jouer une fois – ou PLAY_LOOP – jouer en boucle)

Void Sound.pause()
Met en pause tous les modules.

Void Sound.resume()
Relance la lecture de tous les modules.

Void Sound.stop()
Arrête la lecture de tous les modules.

Void Sound.setPosition(index, position)
Définit la position du curseur de lecture dans un module.
index (Number) : index du module
position (Number) : nouvelle position du curseur

Boolean Sound.isActive()
Retourne true si le lecteur est actif ; sinon, retourne false.

Void Sound.startJingle(index)
Lance la lecture d'un module comme un jingle.
index (Number) : index du volume à jouer

Void Sound.setModVolume(volume)
Définit le volume du module joué.
volume (Number) : nouvelle valeur du volume, comprise entre 0 et 1024

Void Sound.setJingleVolume(volume)
Définit le volume du jingle joué.
volume (Number) : nouvelle valeur du volume, comprise entre 0 et 1024

Void Sound.setModTempo(tempo)
Définit le tempo du lecteur de module.
tempo (Number) : nouvelle valeur du tempo, comprise entre 512 et 2048

Void Sound.setModPitch(ton)
Définit le ton du lecteur de module.
ton (Number) : nouvelle valeur du ton

SFX (effets sonores) :

Void Sound.loadSFX(index)
Charge un SFX dans la mémoire.
index (Number) : index du SFX à charger

Void Sound.unloadSFX(index)
Décharge un SFX de la mémoire.
index (Number) : index du SFX à décharger

Handle Sound.startSFX(index)
Démarre un effet sonore déjà chargé en mémoire. Retourne un handle de cet SFX.
index (Number) : index du SFX à lire

Void Sound.stopSFX(handle)
Arrête un SFX joué.
handle (Handle) : handle d'un SFX, retourné par la fonction startSFX

Void Sound.releaseSFX(handle)
Marque un effet en priorité faible.
handle (Handle) : handle d'un SFX, retourné par la fonction startSFX

Void Sound.stopAllSFX()
Arrête tous les SFX joués.

Void Sound.setSFXVolume(handle, volume)
Définit le volume d'un SFX joué.
handle (Handle) : handle d'un SFX, retourné par la fonction startSFX
volume (Number) : nouvelle valeur du volume, comprise entre 0 et 255 (différente de celle des mods)

Void Sound.setSFXPanning(handle, balance)
Définit la balance d'un SFX joué.
handle (Handle) : handle d'un SFX, retourné par la fonction startSFX
balance (Number) : nouvelle valeur de la balance, comprise entre 0 (gauche) et 255 (droite)

Void Sound.setSFXPitch(handle, ton)
Définit le ton d'un SFX joué.
handle (Handle) : handle d'un SFX, retourné par la fonction startSFX
ton : nouvelle valeur du ton

Void Sound.setSFXScalePitch(handle, gamme)
Définit la gamme de ton d'un SFX joué.
handle (Handle) : handle d'un SFX, retourné par la fonction startSFX
gamme (Number) : nouvelle valeur de gamme du ton

Fichiers INI

Table INI.load(nomFichier)
Charge un fichier INI et crée une table avec son contenu.
nomFichier (String) : fichier à charger

Void INI.save(nomFichier, table)
Sauvegarde une table dans un fichier INI.
nomFichier (String) : fichier dans lequel la table sera sauvegardée
table (Table) : table à sauvegarder

Exemple :
Un fichier INI contient :
```
[infos]
nom=toto
age=25
```

```
table = INI.load('monFichier.ini')
table['infos']['nom'] = 'tata'
INI.save('monFichier.ini', table)
```

Cet exemple change la ligne nom=toto en nom=tata.

Les tables de données des fichiers INI ne peuvent contenir que des Strings !
Si vous voulez sauvegarder des nombres, vous devez d'abord les convertir en String grâce à tostring().

Notes du traducteur :

- ✓ Il n’est plus fait mention du clavier, celui-ci ayant été enlevé des librairies de base de Micro Lua. Vous pouvez toujours trouver une version traduite et opérationnelle sur le forum.
- ✓ Les types de variables n'ont pas été traduits pour des raisons de compréhension générale. En programmation, on parlera plus facilement de « string » que de « chaîne de caractères ». Voici les correspondances françaises des types employés ici :
    - Void : « vide » (utilisé par une fonction qui ne renvoie rien)
    - Number : nombre
    - String : chaîne de caractères
    - Color : couleur
    - Image : image
    - Font : police d'écriture
    - Table : table (ou tableaux)
    - Map & ScrollMap : carte et carte défilante
    - Canvas & CanvasObject : canevas & objet de canevas
    - Constant : constante
    - Boolean : booléen (true – vrai, ou false – faux)
    - DateTime : objet représentant une date
    - Handle : référence d’un objet pour le gérer