Funnel: ActionScript 3 API Reference Manual

Package	Description
funnel	Funnelライブラリの基本パッケージです
funnel.gui	画面上にGUIとして表示するI/Oモジュールのパッケージです
funnel.i2c	I2Cデバイスのパッケージです
funnel.ui	フィジカルUIのパッケージです

パッケージ：funnel

Arduino

ファームウェアとしてFirmataをアップロードしたArduinoをI/Oモジュールとして扱うためのクラスです。

Public Properties

FIRMATA : Configuration	Arduino用のデフォルトのコンフィギュレーションを取得します。戻り値のコンフィギュレーションを変更するにはsetDigitalPinModeを利用します。
gui : IOModuleGUI	画面上で表示するGUIです。

Public Methods

Arduino(config:Configuration = null, host:String = "localhost", portNum:Number = 9000, samplingInterval:int = 33)	コンストラクタ
analogPin(pinNum:uint):Pin	pinNumで指定したアナログピンを取得します。
digitalPin(pinNum:uint):Pin	pinNumで指定したデジタルピンを取得します。
sendFirmataString(stringToSend:String):void	Firmataプロトコルを利用して文字列を送信します
sendSysexMessage(command:uint, message:Array):void	Firmataプロトコルを利用してシステム・エクスクルーシブ・メッセージを送信します
setServoPulseRange(pinNumber:uint, minPulse:uint, maxPulse:uint):void	サーボのパルス幅をマイクロ秒単位でセットします。

Configuration

I/Oモジュールのコンフィギュレーションを設定するためのクラスです。

Public Properties

ainPins : Array	Gainer.analogInputを実際のピン番号に対応させるテーブルです
analogPins : Array	Arduino.analogPinを実際のピン番号に対応させるテーブルです
aoutPins : Array	Gainer.analogOutputを実際のピン番号に対応させるテーブルです
button : uint	Gainer.buttonを実際のピン番号に対応させるテーブルです
config : Array	ピンのタイプ(AIN、DIN、AOUT、DOUT)の配列です
digitalPins : Array	Arduino.digitsalPinを実際のピン番号に対応させるテーブルです
dinPins : Array	Gainer.digitalInputを実際のピン番号に対応させるテーブルです
doutPins : Array	Gainer.digitalOutputを実際のピン番号に対応させるテーブルです
led : uint	Gainer.ledを実際のピン番号に対応させるテーブルです
moduleID : uint	モジュールのID
powerPinsEnabled : Boolean	I2Cデバイス用の電源ピンが有効になっているかどうか
servoPins : Array	サーボを利用できるピンのテーブルです

Public Methods

enablePowerPins():void	I2Cデバイス用の電源ピンを有効にします
setDigitalPinMode(pinNum:uint, mode:uint):void	デジタルピンのモードを設定します。Arduino、Fio、XBee使用時に利用します。

Convolution

入力に対して畳み込み演算を行うクラスです。細かいノイズを取り除くためのローパスフィルタや、ドリフトを取り除くためのハイパスフィタ等があります。

Public Properties

coef : Array	入力バッファの積和を行う際に用いる係数の配列。代入した場合、バッファはクリアされます。

Public Methods

Convolution(kernel:Array)	コンストラクタ
processSample(val:Number):Number	フィルタを適応します
Public Constants	
HPF : Array	ハイパスフィルタのカーネル。コンストラクタに渡すことで利用します。
LPF : Array	ローパスフィルタのカーネル。コンストラクタに渡すことで利用します。
MOVING_AVERAGE : Array	移動平均フィルタのカーネル。コンストラクタに渡すことで利用します。

Fio

FioクラスはFunnel I/Oモジュールを扱うためのクラスです。

Public Properties	
FIRMATA : Configuration	Fio用のデフォルトのコンフィギュレーションを取得します。
Public Methods	
Fio(nodes:Array = null, config:Configuration = null, host:String = "localhost", portNum:Number = 9000, samplingInterval:int = 33)	
Public Constants	
ALL : uint	全てのモジュールを表します。fio.module(ALL).pin(10).value = x のようにすることで、全モジュールの10番目のピンの値をxに設定します。

FunnelErrorEvent

Funnelのエラーイベントを表すクラスです。

Public Constants	
CONFIGURATION_ERROR : String	指定したコンフィギュレーションの設定に失敗したとき送出されます。
ERROR : String	エラーが起きたとき送出されます。
REBOOT_ERROR : String	I/Oモジュールの再起動に失敗したとき送出されます。

FunnelEvent

Funnelのエラーではないイベントを表すクラスです。

Public Constants	
I2C_POWER_PINS_READY : String	I2Cデバイス用の電源ピンが利用可能
READY : String	I/Oモジュールの初期設定が完了して利用可能

Gainer

Gainer I/Oモジュールを扱うためのクラスです。

Public Properties	
button : Button	I/Oモジュール上のボタン
gui : IOModuleGUI	GUI
led : LED	I/Oモジュール上のLED
MODE1 : Configuration	Gainerに用意されている8種類のコンフィギュレーションのうちの最も基本的なものです。アナログ入力：4、デジタル入力：4、アナログ出力：4、デジタル出力：4という構成です。
MODE2 : Configuration	アナログ入力：8、アナログ出力：4、デジタル出力：4
MODE3 : Configuration	アナログ入力：4、デジタル入力：4、アナログ出力：8
MODE4 : Configuration	アナログ入力：8、アナログ出力：8
MODE5 : Configuration	デジタル入力：16
MODE6 : Configuration	デジタル出力：16
MODE7 : Configuration	マトリクスLED

Public Methods	
Gainer(config:Configuration = null, host:String = "localhost", portNum:Number = 9000, samplingInterval:int = 33)	コンストラクタ
analogInput(pinNum:uint):Pin	pinNumで指定したアナログ入力ピンを取得します。
analogOutput(pinNum:uint):Pin	pinNumで指定したアナログ出力ピンを取得します。
digitalInput(pinNum:uint):Pin	pinNumで指定したデジタル入力ピンを取得します。
digitalOutput(pinNum:uint):Pin	pinNumで指定したデジタル出力ピンを取得します。

MatrixLED

Gainer I/Oモジュールに接続したマトリクスLEDをコントロールするクラスです。

Public Methods	
MatrixLED(host:String = "localhost", portNum:Number = 9000)	コンストラクタ
scanMatrix(image:*):void	8x8画素のBitmapDataか、64要素のNumber配列からマトリクスLEDの表示内容を更新します。

Osc

LEDをふわふわ点滅させたりする時などに使います。回数を1回に設定すると、ワンショットの制御にも使えます。サービス間隔はOsc.serviceIntervalの設定に従います。

Public Properties	
amplitude : Number	振幅
freq : Number	周波数
offset : Number	オフセット
phase : Number	位相
serviceInterval : uint	サービス間隔
times : Number	リピート回数（0で無限回）
value : Number	生成された数値
wave : Function	波形の関数

Public Methods	
Osc(wave:Function = null, freq:Number = 1, amplitude:Number = 1, offset:Number = 0, phase:Number = 0, times:Number = 0)	コンストラクタ
IMPULSE(val:Number, lastVal:Number):Number	インパルス
LINEAR(val:Number, lastVal:Number):Number	Linear
reset():void	Oscの状態をリセットします。
SAW(val:Number, lastVal:Number):Number	ノコギリ波
SIN(val:Number, lastVal:Number):Number	サイン波
SQUARE(val:Number, lastVal:Number):Number	矩形波
start():void	Oscの動作を開始します。
stop():void	Oscの動作を停止します。
TRIANGLE(val:Number, lastVal:Number):Number	三角波
update(interval:int = -1):void	指定したインターバルだけ時間を進めます。引数を省略するとOsc.serviceIntervalだけ時間を進めます（ピンに対してフィルタとしてセットした場合には自動的にこの処理が行われます）。

Pin

I/Oモジュールの入出力ピンを表すクラスです。

Public Properties	
average : Number	平均値
filters : Array	ピンに適応するフィルタ配列
lastValue : Number	ピンの変化する前の値
maximum : Number	最大値
minimum : Number	最小値
number : uint	ピン番号
preFilterValue : Number	フィルタをかける前の値
type : uint	ピンのタイプ（AIN、DIN、AOUT、DOUT）
value : Number	センサからの入力値、またはアクチュエータへの出力値

Public Methods	
Pin(number:uint, type:uint)	コンストラクタ
addFilter(newFilter:*):void	フィルタを追加
clear():void	現在までの最大値、最小値、平均値をクリア
removeAllFilters():void	全てのフィルタを削除
setFilters(newFilters:Array):void	複数のフィルタを一度にセット
Public Constants	
AIN : uint	アナログ入力
AOUT : uint	アナログ出力
DIN : uint	デジタル入力
DOUT : uint	デジタル出力
I2C : uint	I2C
PWM : uint	PWM（アナログ出力）
SERVO : uint	サーボ

PinEvent

ピン値の変化に関連するイベントを表すクラスです。

Public Constants	
CHANGE : String	ピンの値が変化
FALLING_EDGE : String	ピンの値が0以外から0に変化
RISING_EDGE : String	ピンの値が0から0以外に変化

Scaler

ある範囲の入力をある範囲にスケーリングするためのクラスです。直線でのスケーリング以外に、よく使われるカーブも用意されています。

Public Properties	
inMax : Number	入力の最大値
inMin : Number	入力の最小値
limiter : Boolean	指定した範囲を超えた入力値を制限するか否か
outMax : Number	出力の最大値
outMin : Number	出力の最小値
type : Function	マッピングに使用する曲線を表す関数

Public Methods	
Scaler(inMin:Number = 0, inMax:Number = 1, outMin:Number = 0, outMax:Number = 1, type:Function = null, limiter:Boolean = false)	コンストラクタ
CUBE(val:Number):Number	y = x^4
CUBE_ROOT(val:Number):Number	y = pow(x, 1/4)
LINEAR(val:Number):Number	y = x
processSample(val:Number):Number	フィルタを適応します
SQUARE(val:Number):Number	y = x^2
SQUARE_ROOT(val:Number):Number	y = sqrt(x)

SetPoint

アナログの値に対して閾値とヒステリシスを持つポイントをセットし、現在の状態を段階化して返します。ポイントが1つの場合の出力は0または1の2種類、ポイントが2つの場合は0または1または2の3種類、ポイントがn個の場合は0からnまでのn種類になります。

Public Methods	
SetPoint(points:Array = null)	コンストラクタ
addPoint(threshold:Number, hysteresis:Number = 0):void	新しいポイントを追加する
processSample(val:Number):Number	フィルタを適応します
removePoint(threshold:Number):void	指定した閾値に設定されているポイントを削除する

SignalScope

入力の状態を表示するためのスコープクラスです。

Public Methods	
SignalScope(left:Number, top:Number, points:int, description:String = "", rangeMin:Number = 0, rangeMax:Number = 1)	コンストラクタ
update(input:*):void	PinまたはNumberを引数として渡してスコープ表示を更新します。

XBee

Digi InternationalのXBeeシリーズを扱うためのクラスです。

Public Properties	
MULTIPOINT : Configuration	XBee 802.15.4用のコンフィギュレーションを取得します。
ZB : Configuration	XBee ZB ZigBee PRO用のコンフィギュレーションを取得します。ただし、D6、D8とD9はモジュールの仕様によりアクセスできません
Public Methods	
XBee(nodes:Array = null, config:Configuration = null, host:String = "localhost", portNum:Number = 9000, samplingInterval:int = 33)	コンストラクタ

パッケージ：funnel.gui

ArduinoGUI

Arduinoボードの入出力をモニタとしてグラフィカルに表示したり、Funnel Serverが起動していない時にシミュレータの役割を果たすクラスです。

Public Methods	
ArduinoGUI()	コンストラクタ

GainerGUI

Gainer I/Oモジュールの入出力をモニタとしてグラフィカルに表示したり、Funnel Serverが起動していない時にシミュレータの役割を果たすクラスです。

Public Methods	
GainerGUI()	コンストラクタ

パッケージ：funnel.ui

Button

フィジカルなボタンを扱うためのクラス

Public Methods	
Button(buttonPin:Pin, buttonMode:uint, longPressDelay:Number = 1000)	コンストラクタ
Public Constants	
PULL_DOWN : uint	コンストラクタにbuttonModeとして与える定数（プルアップ）
PULL_UP : uint	コンストラクタにbuttonModeとして与える定数（プルダウン）

ButtonEvent

フィジカルなボタンで発生するイベントを扱うためのクラス

Public Constants	
LONG_PRESS : String	ボタンが長押しされた
PRESS : String	ボタンが押された
RELEASE : String	ボタンが離された
SUSTAINED_PRESS : String	ボタンの長押しが継続された

LED

LEDを扱うためのクラス

Public Properties	
intensity : Number	明るさ
Public Methods	
LED(ledPin:Pin, driveMode:uint)	コンストラクタ
blink(interval:Number, times:Number = 1, wave:Function = null):void	LEDを指定した間隔、回数、波形で点滅させる
fadeIn(time:Number = 1000):void	指定した時間でフェードイン
fadeOut(time:Number = 1000):void	指定した時間でフェードアウト
fadeTo(to:Number, time:Number = 1000):void	指定した明るさに指定した時間でフェード
isOn():Boolean	LEDがオンか否か
off():void	LEDをオフに
on():void	LEDをオンに
stopBlinking():void	点滅を停止
toogle():void	オン／オフを切り替える
Public Constants	
SOURCE_DRIVE : uint	コンストラクタにderiveModeとして与えるLEDのドライブモード（ソース）
SYNC_DRIVE : uint	コンストラクタにderiveModeとして与えるLEDのドライブモード（シンク）