Documentation by YARD 0.8.5.2

propeller

GUI for propeller display

Run "yard" In the root directory of this app to generate documentation, and then "yard server" to run server with it

Top Level Namespace

Defined Under Namespace

Classes: Preview, Propeller, Rectangle

Constant Summary

X_DIM =
distance from first diode to rotation axis level along diodes line

186.8

Y_DIM =
distance from rotation axis to line containing diodes

23.0

D_DIST =
distance between diodes

4.0

OUTER =
Diodes count including the hole

50

INNER =
Diodes count in the hole

10

Class: Propeller

Inherits:	Object	show all
Defined in:	app/lib/propeller.rb	

Overview

Main class used to communicate between all modules

Defined Under Namespace

Classes: ImageProcessor, Interface, Transmitter, Windows

Instance Method Summary (collapse)

- (Object) **connect_device**(port)
  Connect transmitter with given serial port.
- (Object) **exit**
  Exits the propeller program including:
  - stopping the propeller machine
  - closing the interface.
- (Propeller) **initialize**(args) constructor
  Initializes propeller class ith external processors:.
- (Object) **process_image**(path, placement = {})
  Process image stored in passed url.
- (Object) **process_text**(text, color = "#000000")
  Process text specified by user.
- (Object) **run**(command, args)
  Runs chosen command.
- (Object) **start**
  Starts up the propeller machine.
- (Object) **stop**
  Stops the propeller machine.
- (Object) **transmit**
  Transmits information about selected image to propeller microprocessor.

Constructor Details

- (Propeller) **initialize**(args)

Note: interface - managing data being sent to and from user

Note: propeller_processor - processing image to data readable by robot

Note: preview_processor - generating preview image basing on propeller data

Initializes propeller class ith external processors:

Instance Method Details

- (Object) **connect_device**(port)

Connect transmitter with given serial port

Parameters:

- **port** (String) —
  - port to connect with i.e. /dev/rfcomm0

- (Object) **exit**

Exits the propeller program including:

- stopping the propeller machine
- closing the interface

- (Object) **process_image**(path, placement = {})

Process image stored in passed url

Parameters:

- **path** (String) — path to selected raw image
- **placement** (Hash) *(defaults to: {})* — placement of the image on the propeller display, including Width(w), Height(h), XOffset(x), YOffset(y)

- (Object) **process_text**(text, color = "#000000")

Note: not used by far

Process text specified by user

- (Object) **run**(command, args)

Note: By far it ignores any command

Note: In the future, it should take some kind of communication information as args

Runs chosen command

- (Object) **start**

Starts up the propeller machine

- (Object) **stop**

Stops the propeller machine

- (Object) **transmit**

Transmits information about selected image to propeller microprocessor

Class: Propeller::Windows

Inherits:	Qt::Application	show all
Defined in:	app/lib/propeller/windows.rb	

Overview

Class responsible for displaying windows using QT

Constant Summary

WINDOW_WIDTH =

650

WINDOW_HEIGHT =

500

PREVIEW_WIDTH =

400

PREVIEW_HEIGHT =

400

MARGIN_BIG =

20

MARGIN =

10

BACKGROUND =

"#eeeeee"

IMAGE_PLACEHOLDER =

File.expand_path("../../spec/assets/test.jpg", Pathname(__FILE__).dirname.realpath)

Instance Method Summary

(collapse)

- (Object) **center_window** private

  Centers window main window.

- (Object) **change_image** private

  Triggered on change image button click.

- (Object) **change_text** private

  Triggered on load text button click.

- (Object) **connect_device** private

Triggered on connect device button click.

- (Object) **init_gui_elements** private
  Sets GUI interface.

- (Windows) **initialize**(args, interface) constructor
  A new instance of Windows.

- (Object) **load_preview**(path)
  Loads image to preview part.

- (Object) **pick_color** private
  Open dialog to choose text color.

- (Object) **send_data** private
  Triggered on send data button click.

Constructor Details

- (Windows) **initialize**(args, interface)

A new instance of Windows

Instance Method Details

- (Object) **center_window** (private)

Centers window main window

- (Object) **change_image** (private)

Triggered on change image button click

- (Object) **change_text** (private)

Triggered on load text button click

- (Object) **connect_device** (private)

Triggered on connect device button click

- (Object) **init_gui_elements** (private)

Sets GUI interface

- (Object) **load_preview**(path)

Loads image to preview part

Parameters:

- **path** —
  - Path to preview image

```
- (Object) pick_color (private)
```

Open dialog to choose text color

```
- (Object) send_data (private)
```

Triggered on send data button click

Class: Preview

Inherits:	Object	show all
Defined in:	app/lib/propeller/preview.rb	

Overview

Class userd to generate preview based on propeller data

Instance Method Summary

(collapse)

- – (String) **generate**(pixels, radius = 200)
  Generates preview image based on propeller data.
- – (Preview) **initialize** constructor
  A new instance of Preview.
- – (Object) **multicolor_preview**
  Method used to generate some stripped pattern.

Constructor Details

```
- (Preview) initialize
```

A new instance of Preview

Instance Method Details

```
- (String) generate(pixels, radius = 200)
```

Generates preview image based on propeller data

Parameters:

- **pixels** (Array) — Matrix of pixels generated to be displayed on robot
- **radius** (Integer) *(defaults to: 200)* — Size of output image

Returns:

- (String) — Absolute path to preview image

- (Object) **multicolor_preview**

Note: this method is not used anywhere, but maybe useful later on

Method used to generate some stripped pattern

Class: Rectangle

Inherits:	Qt::Widget	show all
Defined in:	app/lib/propeller/rectangle.rb	

Overview

QT widget used to generate rectangle for colorPicker

Instance Attribute Summary (collapse)

- (Object) **color** writeonly
  Sets the attribute color.

Instance Method Summary (collapse)

- (Rectangle) **initialize**(parent = nil, color = nil) constructor
  TODO yard.
- (Object) **paintEvent**(event)
  TODO yard.
- (Object) **update_color**(color = nil)
  TODO yard.

Constructor Details

- (Rectangle) **initialize**(parent = nil, color = nil)

TODO yard

Parameters:

- **parent** (?) *(defaults to: nil)*
- **color** (Color) *(defaults to: nil)* — color to display on preview

Instance Attribute Details

- (Object) **color**=(value) (writeonly)

Sets the attribute color

Parameters:

- **value** — the value to set the attribute color to.

Instance Method Details

- (Object) **paintEvent**(event)

TODO yard

Parameters:

- **event** —

- (Object) **update_color**(color = nil)

TODO yard

Parameters:

- **color** (Color) *(defaults to: nil)*

Class: Propeller::Interface

Inherits:	Object	show all
Defined in:	app/lib/propeller/interface.rb	

Overview

class used as interface for windows interface, allowing communication with propeller class

Instance Method Summary

(collapse)

- (Object) **connect_device**(path)
  Connects with serial port.
- (Object) **hide**
  Hides the interface.
- (Object) **hide_loader**
  Hides loading animation and unlocks UI.
- (Interface) **initialize**(args, propeller) constructor
  A new instance of Interface.
- (Object) **processed**(path)

Shows preview and unlocks UI actions.

- (Object) **reload_image**(path, placement = {})
  Orders propeller to start processing image selected by user including propeller data generation and preview rendering.

- (Object) **reload_text**(text, color)
  pending.

- (Object) **send_data**
  Start sending data to propeller device.

- (Object) **show**
  Shows the interface.

- (Object) **show_loader**
  Shows loading animation and blocks UI.

- (Object) **show_preview**(path)
  Displays preview of processed preview image stored at path.

Constructor Details

- (Interface) **initialize**(args, propeller)

A new instance of Interface

Instance Method Details

- (Object) **connect_device**(path)

Connects with serial port

Parameters:

- **path** (String) —
  - deivce to connect with

- (Object) **hide**

Hides the interface

- (Object) **hide_loader**

Hides loading animation and unlocks UI

- (Object) **processed**(path)

Shows preview and unlocks UI actions

Parameters:

- **path** (String) — path to processed preview image

- (Object) **reload_image**(path, placement = {})

Orders propeller to start processing image selected by user including propeller data generation and preview rendering

Parameters:

- **path** (String) — path to image specified by user
- **placement** (Hash) *(defaults to: {})* — pacement hash specified by user, including X offset(x), Y offset(y) and diameter size (s)

- (Object) **reload_text**(text, color)

pending

- (Object) **send_data**

Start sending data to propeller device

- (Object) **show**

Shows the interface

- (Object) **show_loader**

Shows loading animation and blocks UI

- (Object) **show_preview**(path)

Displays preview of processed preview image stored at path

Parameters:

- **path** (String) — path to image (processed for propeller)

Class: Propeller::Transmitter

Inherits:	Object	show all
Defined in:	app/lib/propeller/transmitter.rb	

Overview

Class responsible for data transmition between propeller device and application

Constant Summary

SPEED =

Data transmition speed in bits / second

115200

Instance Method Summary

(collapse)

- (Array) **compress**(data) private
  Compress array of r,g,b values to 8bit values ready to send to propeller.
- (Object) **connect**(port)
  Connects with specified serial port.
- (Array) **convert**(data) private
  Convert 2d array of pixels to flatten thirds of r,g,b values of each pixel in.
- (Object) **disconnect**
  Disconnect serial port.
- (Object) **transmit**(pixels)
  Transmit image given in 2d array to propeller using chosen serial port.
- (Object) **upscale**(value) private
  Upscales value from 8 bit to 12 bit.

Instance Method Details

- (Array) **compress**(data) (private)

Compress array of r,g,b values to 8bit values ready to send to propeller

Parameters:

- **data** (Array) —
  - array of decimal r,g,b values

Returns:

- (Array) —
  - array of 8bit values ready to send to propeller

- (Object) **connect**(port)

Connects with specified serial port

Parameters:

- **port** (String) —
  - Serial port descriptor i.e /dev/rfcomm0

- (Array) **convert**(data) (private)

Convert 2d array of pixels to flatten thirds of r,g,b values of each pixel in

Parameters:

- **data** (Array[Array]) —

- Array of image pixel values in rgba format

Returns:

- (Array) —
  - Flatten thirds of decimal r,g,b values

- (Object) **disconnect**

Disconnect serial port

- (Object) **transmit**(pixels)

Transmit image given in 2d array to propeller using chosen serial port

Parameters:

- **pixels** (Array[Array]) —
  - array of hexadecimal pixels values

- (Object) **upscale**(value) (private)

Upscales value from 8 bit to 12 bit

Parameters:

- **value** (Integer)

Class: Propeller::ImageProcessor

Inherits:	Object show all
Defined in:	app/lib/propeller/image_processor.rb

Overview

Main image processor, converting image to matrix of pixels to display on robot

Constant Summary

PLACEMENT =
Default placement hash

{x: 0, y: 0, s: 200}

Instance Method Summary (collapse)

- (Array) **compute_radius**(radius)
  Get all pixels on the circle of radius 'radius' to display on propeller.

- (Object) **crop_square**

Crops the square part of the image in selected position given on @placement x - offset on x axis, y - offset on y axis, s - diameter of the displayed circle.

- (Object) **depolarize**

Decompose image to angular data taking center pixel as rotation axis.

- (ImageProcessor) **initialize** constructor

A new instance of ImageProcessor.

- (Array) **process**(original_path, placement)

Change image to format readable by propeller display.

- (Object) **read_row**(radius)

Reads nth row in image and convert it to array of pixels.

- (Object) **resize**

Resize image to dimensions necessary for propeller display, so each pixel will match one led.

- (Object) **rotate**(angle)

Rotates the image in order to adjust the corresponding diods offset.

Constructor Details

```
- (ImageProcessor) initialize
```

A new instance of ImageProcessor

Instance Method Details

```
- (Array) compute_radius(radius)
```

Get all pixels on the circle of radius 'radius' to display on propeller

Parameters:

- **radius** (Integer) — index of diode to fetch row for

Returns:

- (Array) — array of pixels - one pixel for each angle for given diode

---

```
- (Object) crop_square
```

Crops the square part of the image in selected position given on @placement x - offset on x axis, y - offset on y axis, s - diameter of the displayed circle

---

```
- (Object) depolarize
```

Decompose image to angular data taking center pixel as rotation axis

---

```
- (Array) process(original_path, placement)
```

Change image to format readable by propeller display

Parameters:

- **original_path** (String) — path to orginal image
- **placement** (Hash) — hash with info about dimensions and offset

Returns:

- (Array) — matrix of pixels to display by propeller [distance]

- (Object) **read_row**(radius)

Reads nth row in image and convert it to array of pixels. Each pixes is represented as hex RGBA

Parameters:

- **radius** (Integer) — Index of pixel row

- (Object) **resize**

Resize image to dimensions necessary for propeller display, so each pixel will match one led

Examples:

For example if propeller takes 360 angles on 40 diods with outer radius 50 (indluding hole) it resizes it to 360x50

- (Object) **rotate**(angle)

Rotates the image in order to adjust the corresponding diods offset

Parameters:

- **angle** (Float) — angle in radians. Image is rotated CCW
