Porthos

An-embedded-linux-robot

0.0.0-cmake

generated on Sun Nov 20 2016 00:21:03

Contents

Chapter 1

Porthos

This project tries to create an embedded linux robot system.

A high-level description of the system is given in the system_description. Some requirements have been created, but these are still very much open to discussion.

The compiled documentation can be read at `http://spoorcc.github.io/porthos/` And the PDF at `http://spoorcc.github.io/porthos/Porthos.pdf`

Compiling

```
mkdir bld
cd bld
cmake ..
make
```

Generating documentation

```
cd bld
make doc
```

Chapter 2

Licensing

This page describes the licencing for the Porthos system.

2.1 Licensing of documentation

Todo determine licensing

2.2 Licensing of source code

Todo determine licensing

Chapter 3

System Description

Describes the robot system on high-level

Chapter 4

Testing

This page describes the testing procedures for the Porthos project.

4.1 Unittests

This project uses check for testing the C-code In order to run the tests do the following

```
cd bld
cmake ..
make
make test
```

To have more output for analyzing failing tests use follwoing command instead of make test:

```
ctest --verbose
```

Chapter 5

Workflow

This page describes the workflow for the Porthos project.

5.1 Workflow

This project works following the git-flow branching model. Each feature is developed on a feature branch, branched of of develop. Check out `http://nvie.com/posts/a-successful-git-branching-model/` for more info.

The below workflow is based on `http://qq.is/tutorial/2011/10/23/git-flow-on-github.↩ html`

5.2 Setting up

First clone the repository

```
git clone https://github.com/spoorcc/porthos.git
```

Go into the repo

```
cd porthos
```

Setup the origin

```
git remote add upstream git@github.com:spoorcc@porthos
```

Setup git flow (first install git flow if you haven't got it)

```
git flow init
```

And accept all the defaults

5.3 Starting on your feature

Create a new branch for your awesome feature

```
git flow feature start <my_great_feature>
```

Push the branch remote.

```
git flow feature publish <my_great_feature>
```

Commit your changes reguraly locally with descriptive messages.
Also push the changes back up to GitHub.

```
git push origin HEAD
```

5.4 Finish work

Create a pull request in the GitHub interface. In the pull request add usefull info. Click the send pull request to confirm you think you're done.

When your awesome feature is reviewed, sometimes additional changes are needed. Make them locally, commit and push them up to your branch.
Make sure your on your feature branch:

```
git checkout feature/<my_awesome_feature>
```

Do your development, commit and push the changes again. (see Starting on your feature).

5.5 Cleanup

When all your changes are agreed upon and merged by the project, your feature branch will be deleted. Locally you can finish your feature as well.

```
git flow feature finish
```

Chapter 6

Todo List

Page Licensing

determine licensing

determine licensing

Chapter 7

Class Index

7.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

Chapter 8

File Index

8.1 File List

Here is a list of all files with brief descriptions:

Chapter 9

Class Documentation

9.1 position_t Struct Reference

```
#include <motion.h>
```

Public Attributes

- double x
- double y

9.1.1 Detailed Description

Simple position struct

9.1.2 Member Data Documentation

9.1.2.1 x

```
double position_t::x
```

9.1.2.2 y

```
double position_t::y
```

The documentation for this struct was generated from the following file:

- motion.h

Chapter 10

File Documentation

10.1 licensing.dox File Reference

10.2 motion.c File Reference

```
#include <stdio.h>
#include "motion.h"
```

Functions

- int motion_init ()

  Initializes the motion library.
- int motion_get_current_position (position_t ∗position)

  Returns current position.

10.2.1 Function Documentation

10.2.1.1 motion_get_current_position()

```
int motion_get_current_position (
            position_t ∗ position )
```

Returns current position.

10.2.1.2 motion_init()

```
int motion_init ( )
```

Initializes the motion library.

The motion library is initialized and ready to use.

10.3 motion.h File Reference

```
#include <stdio.h>
```

Classes

- struct position_t

Enumerations

- enum MotionError { MOTION_OK = 0, MOTION_PARAMETER_ERROR }

Functions

- int motion_init ()

  Initializes the motion library.
- int motion_get_current_position (position_t ∗position)

  Returns current position.

10.3.1 Enumeration Type Documentation

10.3.1.1 MotionError

```
enum MotionError
```

error code for library

Enumerator

MOTION_OK	Everything went OK
MOTION_PARAMETER_ERROR	Wrong parameter was provided

10.3.2 Function Documentation

10.3.2.1 motion_get_current_position()

```
int motion_get_current_position (
             position_t ∗ position )
```

Returns current position.

10.3.2.2 motion_init()

```
int motion_init ( )
```

Initializes the motion library.

The motion library is initialized and ready to use.

10.4 README.md File Reference

10.5 system_description.dox File Reference

10.6 test_libmotion.c File Reference

```
#include <check.h>
#include <stdio.h>
#include "motion.h"
```

Functions

- void test_init (void)

  Test initialization.
- void test_get_current_position_GW001 (void)

  Test getting position.
- void test_get_current_position_BW001 (void)

  Test getting position.
- Suite ∗ motion (void)
- int main (int argc, char ∗argv[ ])

10.6.1 Function Documentation

10.6.1.1 main()

```
int main (
            int argc,
            char * argv[ ] )
```

10.6.1.2 motion()

```
Suite* motion (
            void  )
```

10.6.1.3 test_get_current_position_BW001()

```
void test_get_current_position_BW001 (
            void  )
```

Test getting position.

Test initialization of Motion library

10.6.1.4 test_get_current_position_GW001()

```
void test_get_current_position_GW001 (
            void  )
```

Test getting position.

Test initialization of Motion library

10.6.1.5 test_init()

```
void test_init (
            void  )
```

Test initialization.

Test initialization of Motion library

10.7 testing.dox File Reference

10.8 workflow.dox File Reference

Index