Table of Contents

1. Willy Git setup

Commands used to set up the git repos in willy.

```
cd /home/willy/Documents
git clone https://github.com/ArtOfRobotics/WWEB
cd WWEB
git checkout -b test
git push -u origin test
-   Username: willythegarbagedisposal@gmail.com
-   Password:
cd ..
git clone https://github.com/ArtOfRobotics/WTGD
cd WTGD
git checkout -b test
git push -u origin test
-   Username: willythegarbagedisposal@gmail.com
-   Password:
```

1.1. Github synchronisation

A crontab has been created which ensures that Willy stays up-to-date with Github. Because Willy does not have a fixed external-IP or domain, it is unfortunately not possible to use Github Webhooks. That is why a crontab job has been set up, which is configured as follows:

```
sudo crontab -e

*/1 * * * * su -s /bin/sh -c '/home/willy/Documents/WTGD/bin/gitsync.sh ' >>
/home/willy/Documents/logs/git.log 2>&1
```

The script used for syncing github can be found on Github. The main purpose of the script is to do the following:

- Update the WTGD code for driving
- Set permissions for script folder
- (In the future): automatic building
- Update the WWEB code for web platform
- Set permisions for specific folders

Currently this line creates a pull of the test branch, later it is desirable to change this to the master branch for production code. Logs can be found in: /home/willy/Documents/logs/git.log

To track the status of git sync live use the following command

```
tail -f /home/willy/Documents/logs/git.log
```

1.2. Github Repo Willy

Can be found in /home/willy/Documents/WTGD

1.3. Github Repo Web platform

Can be found in /home/willy/Documents/WWEB

1.4. Build

Web interface For building the web interface refer to the Development Guide of the Web section.

Willy For building the software used by Willy refer to the Development Guide of the Technical section.

1.5. Accounts

All the accounts coupled to the email adress: willythegarbagedisposal@gmail.com can be found on the Google Drive

References

- link: https://artofrobotics.github.io/WillyWiki/Documents/#_willy_git_setup

2. Operation system of Willy

2.1. Configuration

Currently, Willy is operating using ‘Linux Ubuntu’ as operation system. The ROS-framework is used for centralized communication between nodes.

For more information about nodes and the working of it, see our ROS Introduction wiki page.

Nodes are referred as different software/hardware components of Willy: think of; GPS, sensors, compass and software functions. Some nodes may require

```
dependent-ROS-packages
```

To execute and compile software nodes, these packages are required. Both ROS and dependencies require the same version. The current version of ROS is

```
‘ROS-kinetic’
```

2.2. Current installed packages

The first step in updating the OS is to determine which packages and dependencies are installed. Because previous teams made a lot of changes to Willy, not all Linux packages may still be required. To list the manually installed packages, the following command was used.

```
_comm -23 <(apt-mark showmanual | sort -u) <(gzip -dc
/var/log/installer/initial-status.gz | sed -n 's/^Package: //p' | sort
-u)_
```

The following packages where manually installed;

```
brightness-controller
dhcpcd5
dotnet-sdk-2.0.0
git
google-chrome-stable
htop
nmap
openssh-server
pgadmin3
postgresql
python-pip
ros-kinetic-desktop-full
ros-kinetic-joystick-drivers
ros-kinetic-rosbridge-server
ros-kinetic-rosserial-python
ros-kinetic-move-base
ros-kinetic-hector-mapping
ros-kinetic-sicktim
ros-kinetic-opencv
ros-kinetic-openni
ros-kinetic-rosserial-server
ros-kinetic-teleop-twist-joy
ros-kinetic-teleop-twist-keyboard
ros-lunar-catkin
screen
vsftpd
x11vnc
xfce4
xrdp
```

Some of the above packages are required to compile and execute the WTGD code that is available from GIT. These packages are listed bold. Other packages may be required for the web platform or may have another goal than compiling and executing ROS code. In this project there will be major changes in the web platform, because some of the code will be changed. Dependencies will be determined during the development of the code. Other packages are explained in documentation that will be available with the final delivery.

2.3. Determined packages for Ubuntu 16.04 and ROS-kinetic

To create a clear view about the current 'WTGD' code that Willy contains, and how this works on Ubuntu 16.04 with ROS-Kinetic, a test environment was created. None of the previous listed packages where installed, only Ubuntu 16.04 was installed and the ROS-framework on top of this. The code was not able to build successful in this test environment, however based on the error messages, dependencies where determined. Every time a dependency was missing, the error massage was inspected. We concluded that the following packages are required to execute the 'WTGD' code on Willy.

```
ROS-kinetic-desktop-full
Screen
ROS-kinetic-rosserial
ROS-kinetic-rosserial-arduino
ROS-kinetic-rosapi
```

2.4. Install dependencies

To install above dependencies, the following commands are required.

Install ROS:

```
sudo sh -c 'echo "deb http://packages.ros.org/ros/ubuntu $(lsb_release
-sc) main" > /etc/apt/sources.list.d/ros-latest.list'
```

```
sudo apt-key adv --keyserver hkp://ha.pool.sks-keyservers.net:80
--recv-key 421C365BD9FF1F717815A3895523BAEEB01FA116
```

```
sudo apt-get update
sudo apt-get install -y ros-kinetic-desktop-full
sudo rosdep init
rosdep update
```

Link the ROS framework to the ubuntu bash:

```
echo "source /opt/ros/kinetic/setup.bash" >> ~/.bashrc
source ~/.bashrc
```

Start a new bash prompt and run (test):

```
roscore
```

Install ROS-dependencies:

```
sudo apt-get install Screen
sudo apt-get install ros-kinetic-rosserial
sudo apt-get install ros-kinetic-rosserial-arduino
sudo apt-get install ros-kinetic-rosapi
sudo apt-get install ros-kinetic-rosbridge-server
```

Give user permission to access USB ports:

```
sudo usermod -a -G dialout willy
```

With the above installation changes made to Ubuntu, the current WTGD code was able to run.

2.5. Devices network

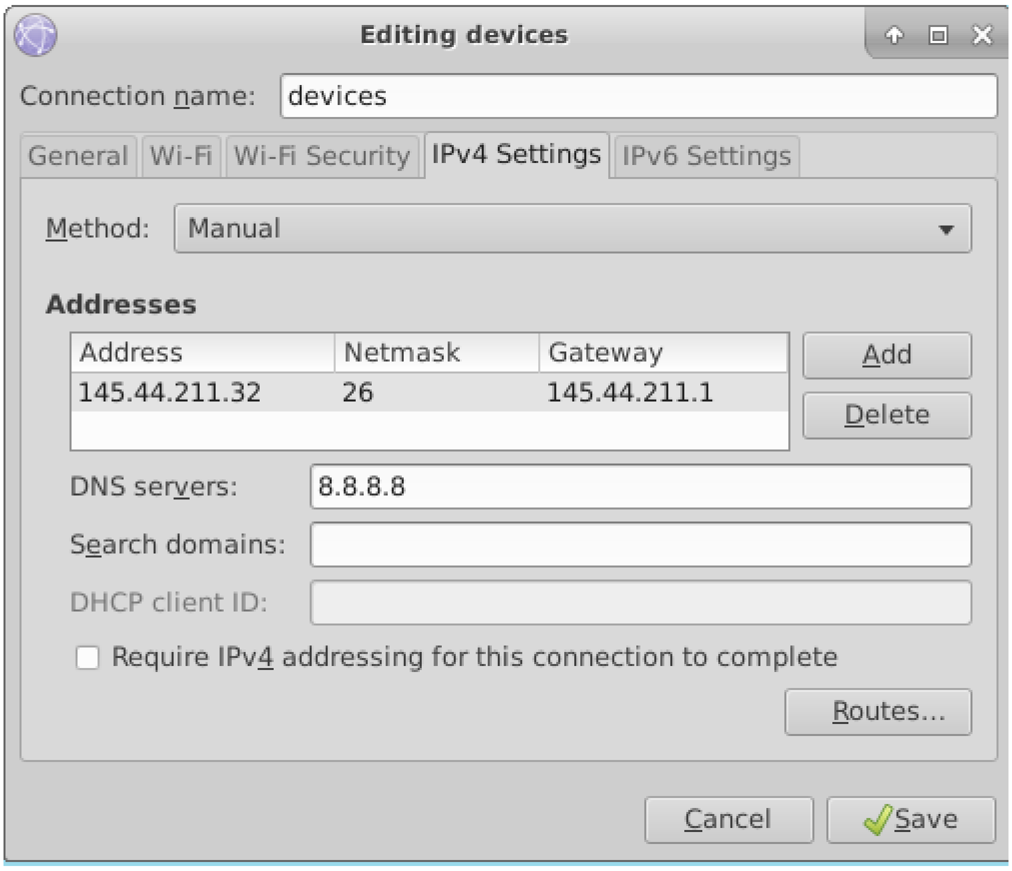
The following information can be used to connect with the 'devices' wifi-network at Windesheim.

```
IP-adres: 145.44.211.32
Subnet mask: 255.255.255.192
Default gateway: 145.44.211.1
DNS: 8.8.8.8
```

An connection can only be made with the following MAC address. If this is changed, please ask the support-desk of Windesheim to change the MAC-address.

MAC: 10:4a:7d:21:f2:d2

The configuration on Willy is as shown below



Connection name: devices

Method: Manual

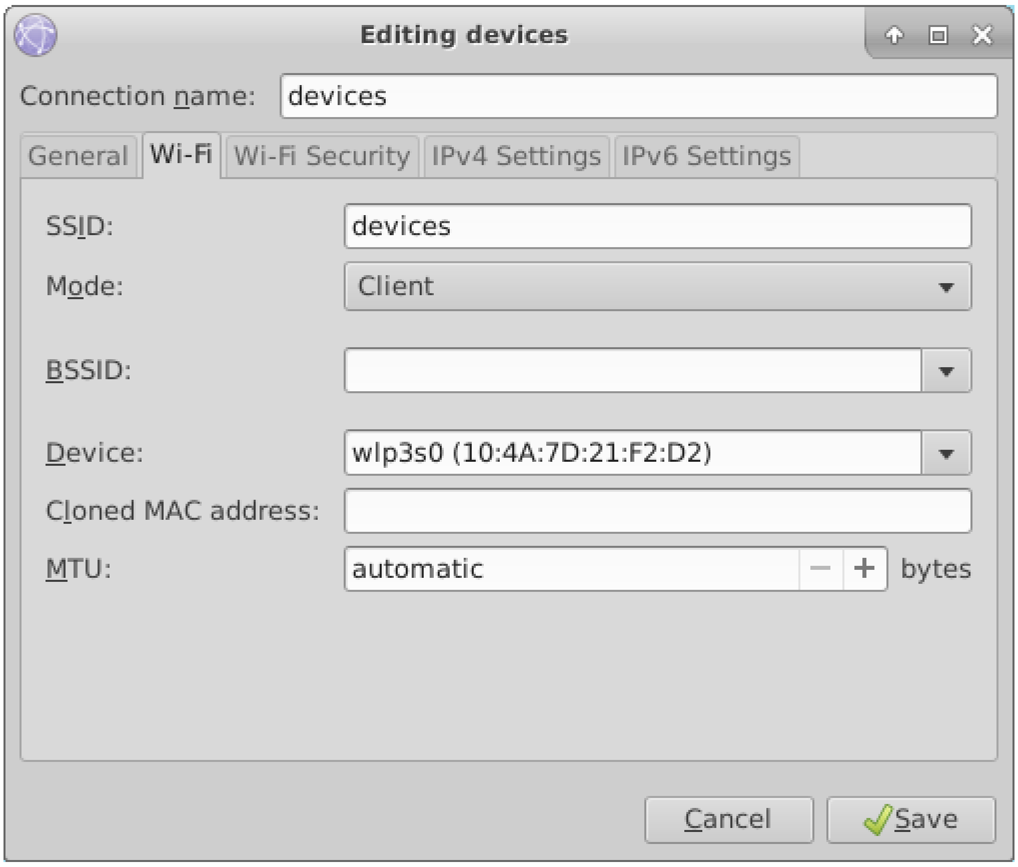
Address	Netmask	Gateway
145.44.211.32	26	145.44.211.1

DNS servers: 8.8.8.8

Search domains:

DHCP client ID:

☐ Require IPv4 addressing for this connection to complete



Connection name: devices

SSID: devices

Mode: Client

BSSID:

Device: wlp3s0 (10:4A:7D:21:F2:D2)

Cloned MAC address:

MTU: automatic bytes

2.6. Leap

Download the Leap package from https://www.leapmotion.com/setup/desktop/linux

Save the package and unzip the 64-bit version

Go to the folder the .deb was unzipped to and run `sudo dpkg --install Leap-*-x64.deb`

3. Remote access

3.1. RDP

The following configuration was used when RDP was deployed by using XRDP and XFCE4

```
sudo apt-get install xrdp
sudo apt-get install xfce4
```

Edit .Xsesion file in home directory

```
echo xfce4-session >~/.xsession
```

Edit XRDP configfile: “/etc/xrdp/starwm.sh” for using XFCE4

```
#!/bin/sh
if [ -r /etc/default/locale ]; then
  . /etc/default/locale
export LANG LANGUAGE
fi
startxfce4
```

Fix the Tab button by editing:

```
~/.config/xfce4/xfconf/xfce-perchannel-xml/xfce4-keyboard-shortcuts.xml
```

Replacing

```
<property name="&lt;Super&gt;Tab" type="string" value="switch_window_key"/>
```

By

```
<property name="&lt;Super&gt;Tab" type="string" value="empty"/>
```

3.2. Openssh server

Install openssh

```
sudo apt-get install openssh-server
```

Enable ssh on boot

```
sudo systemctl enable ssh
```

References

- link: https://artofrobotics.github.io/WillyWiki/Archive/Research/Alternative-interaction.adoc

Status and Advice

- Status
- Todo & Advice

Archive

- (2016/2) Initial design
- (2017/1) Base & Functionalities
- (2017/2) Research

[Build Status]

4. Willy Wiki

This wiki is setup to increase the transferability of the project. Everything you need to start is documented here. Detailed documents are referenced trough the wiki if you need more information.

4.1. Introduction

The wiki is setup using AsciiDoc, TravisCI & Github pages. In practice we use AsciiDoc as source code, TravisCI to convert to html/pdf and Github Pages for publishing the website. This is visually shown in the image below, this is taken from the tutorial we followed.

[AsciiDoc to Github Pages with Travis and Docker Asciidoctor] |

http://mgreau.com/posts/images/cover-asciidoc-ghpages.svg

4.2. Why AsciiDoc

Why AsciiDoc is chosen as our markdown language. During the search for a good Wiki tool, we eventually stumbled upon Github Pages. Github pages in intended to automatically publish markdown for you as a html site and you can add more functions by Using Jekyll. However quickly limitations in Markdown where found and the Jekyll implementation made if far more complex due to different plugins. That's why it is decided to use AsciiDoc at it's raw form.

This is directly the main advantage of using Markdown, while you can use plugins it's main functionality makes it the perfect language for creating a Wiki. In terms of markdown languages you can follow this list as a rule of thumb:

- Markdown (MD)
    - Is the most simplest markdown language out there, but is also it's main weakness
- AsciiDoc (Adoc)
    - Is more versatile in the basics and much more rich in terms of formatting and plugins
- LaTex (Tex)
    - Is more professional focussed and contains a lot of functions at its core where in other markdown languages you need plugins

This should also clarify the reason that AsciiDoc is chosen as the source language of this Wiki. [Markdown vs AsciiDoc]

4.3. How the Wiki is setup using AsciiDoc with TravisCI and Github Pages

The wiki is setup fairly easy, especially when you know your way around Github and TravisCI. So it is important to read into these topics if you don't know what these tools mean. And for AsciiDoc you'll learn it along the way as we did.

4.4. Conversion

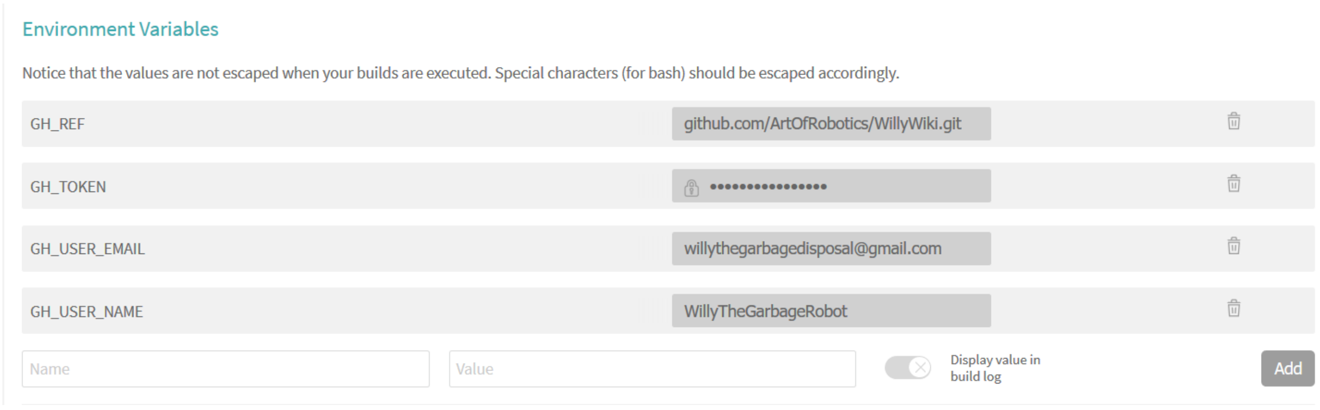
As told in the [introduction] AsciiDoc is used as a source language, which then can be converted to whatever format you like. Most commonly HTML and PDF.

4.4.1. Travis

Travis is setup to convert all documents recursively to HTML and to PDF.

Setup

Before you can use Travis you must give travis acces to the repository, this is already done using Willy's Github account. The following environment variables must be set for the script to work properly.


Environment Variables

Notice that the values are not escaped when your builds are executed. Special characters (for bash) should be escaped accordingly.

GH_REF	github.com/ArtOfRobotics/WillyWiki.git
GH_TOKEN	••••••••••••••••
GH_USER_EMAIL	willythegarbagedisposal@gmail.com
GH_USER_NAME	WillyTheGarbageRobot

Config

The config file used by TravisCI is travis.yml. If you are not familiar with Travis it basically asks you for the following:

- Some config elements (Setup)
  - as of what kind of acces methods and what services you'd like to use
- What to do before installation (Before)
  - Defined under before_installation
- What script to run (Execution)
  - Defined under script:
- What to do after the script has run (After)
  - Defining after_error:, after_failure:, after_success:

For this script a Docker container is used specially made for Asciidoctor, the tool used for AsciiDoc conversion. You do not need any docker knowlegde to use this script because it uses a readymade Docker container. In this container the asciidoctor commands are executed.

Currently the Travis Config is setup as follows:

Setup:
Use of docker and sudo acces

```
sudo: required
```

services: - docker **Before:**
Make a output directory and pull the readymade Docker container.

```
before_install:
  - mkdir -p output
  - docker pull asciidoctor/docker-asciidoctor
```

Execution:
Here is where all the documents are converted what basically comes down to this command:

```
asciidoctor -a allow-uri-read *.adoc
```

With docker it looks like this in the travis.yml

```
script:
  - docker run -v $TRAVIS_BUILD_DIR:/documents/ --name asciidoc-to-html
asciidoctor/docker-asciidoctor asciidoctor -a allow-uri-read **/*.adoc
  - docker run -v $TRAVIS_BUILD_DIR:/documents/ --name asciidoc-to-html-root
asciidoctor/docker-asciidoctor asciidoctor -a allow-uri-read *.adoc
```

Because these commands do not allow for recursive generation more than one folder deep some more lines are added to make sure the Archive folder is converted. A better fix stil need to be implemented.

If the build is failing probably a root heading is used somewhere. This can cause conflicts with the sidebar configuration and is only used in the welcome document.

```
= This is a root heading
== This heading should be used troughout the wiki for the main chapters
```

After:
Your general logging

```
after_error:
  - docker logs
after_failure:
  - docker logs
```

The publishing to Github Pages

```
after_success:
  - find . -name '*html' | cpio -pdm output ;
  - find . -name '*png' | cpio -pdm output ;
  - cd output ;
  - git push
```

Here the output folder contains all the converted documents in html files, but still the images need to be copied, else the images would not be shown. Everything that is copied into the output folder is then pushed to the gh-pages branch on GitHub. As you can see in the travis.yml used some more actions are done by Travis to ensure everything works properly.

For the sidebar to scale correctly we had to manualy add the toc classes because we do not use Docbook (yet). Also see [Recursive replace].

```
  - find -name "*.html" -exec sed -i 's/class="article"/class="article
toc2 toc-left"/g' {} +
```

For conversion from Word to AsciiDoc you can use Pandoc to convert word documents or any other format to AsciiDoc fairly easy. The following command can then be used after Pandoc is installed:

```
pandoc -f "input.docx" "output.adoc"
```

To convert all documents recursively in the current folder you can use the following script: (Windows) [source, BATCH

```
for /r %%v in (*.docx) do pandoc -f "%%v" "%%v.adoc"
```

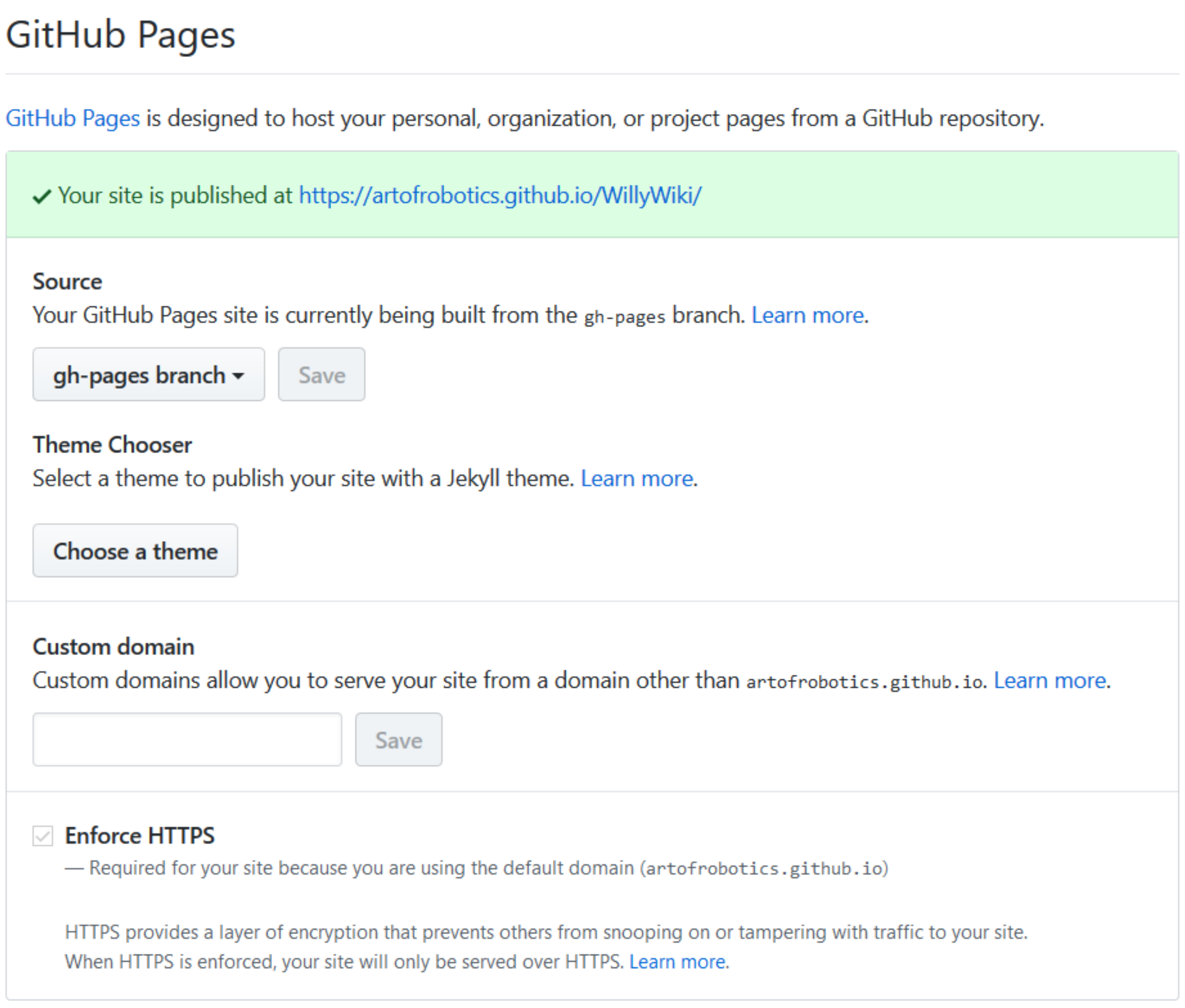
4.5. Publishing

The end result of the Travis script is a folder with html files which can then be hosted on any server even offline.

4.5.1. Github Pages

To do this we use Github Pages, as this is a free service and is perfect for hosting a static html site. It also helps keeping a history of changes.

This is configured as following: . Go to Github, WillyWiki Settings page . Scroll down to Github Pages . Set branch to 'gh-pages' and you are done



GitHub Pages

GitHub Pages is designed to host your personal, organization, or project pages from a GitHub repository.

✓ Your site is published at https://artofrobotics.github.io/WillyWiki/

Source
Your GitHub Pages site is currently being built from the `gh-pages` branch. Learn more.

Theme Chooser
Select a theme to publish your site with a Jekyll theme. Learn more.

Custom domain
Custom domains allow you to serve your site from a domain other than `artofrobotics.github.io`. Learn more.

✓ **Enforce HTTPS**
— Required for your site because you are using the default domain (`artofrobotics.github.io`)

HTTPS provides a layer of encryption that prevents others from snooping on or tampering with traffic to your site.
When HTTPS is enforced, your site will only be served over HTTPS. Learn more.

4.6. Further reading

The Asciidoctor wiki is a good source, you also might notice the familiar look https://asciidoctor.org/docs/user-manual/#introduction-to-asciidoctor

A hardcoded sidebar is used to avoid using Docbook, it might be worth to take a look at this format as it is being used by the official sources as well. The main disadvantage of this is that it makes the Wiki one long HTML page with a large index, however for PDF export this would be fabiolous. See Asciidoctor Wiki as an example.

References

- [Tutorial] Maxime Gréau. Convert AsciiDoc to HTML/PDF & publish to GitHub Pages with Travis CI and Asciidoctor Docker containers
- [[[Markdown vs AsciiDoc]]] Asciidoctor, Markdown vs AsciiDoc
- [[[Recursive replace]]] Stackoverflow, Recursive replace