

Table of Contents

1. Development Guide	2
1.1. Prerequisite	2
1.2. Github	3
1.3. Compilation	4
1.4. Testing/Debugging	4
1.5. Running Scripts	5

Welcome

Project Willy

- [History of Willy](#)
- [Project Willy](#)
- [Publicity](#)
- [Sponsors](#)

Getting started

- [Development Guide](#)
- [Driving Willy](#)
- [Documentation](#)

Build of Willy

- [Design history](#)
- [Requirements](#)
- [Design reference](#)
- [Physical build](#)
- [Hardware](#)

Robotic Operating System

- [Introduction to ROS](#)
- [ROS Tutorials](#)
- [Multi master](#)

Architecture

- [Software Architecture](#)
- [Hardware Architecture](#)
- [Skylab Architecture](#)
- [ROS topic design](#)

Hardware nodes

- [sensor node](#)
- [si node](#)
- [power node](#)
- [WillyWRT](#)

Components

- [ROS master](#)
- [New ROS master on Ubuntu](#)
- [Brain](#)

- [Sonar](#)
- [Lidar](#)
- [Localization and navigation](#)
- [Motor controller](#)
- [Joystick](#)
- [Social interaction](#)
- [Speech](#)
- [Speech recognition](#)

Skylab

- [Setup Skylab](#)
- [Python scripts](#)
- [Webserver](#)
- [Functions of the webserver](#)
- [Skylab servers](#)
- [ROS installation on Ubuntu VMs in Skylab](#)
- [DNS,DHCP, pfSense & Ubuntu](#)

Radeffect App

- [Radeffect App](#)

Lessons learned

- [Todo & Advice](#)
- [Lessons Learned](#)

Archive

- [Previous Groups](#)
- [Research Archive](#)

1. Development Guide

1.1. Prerequisite

- Basic knowlegde of NodeJs, Git and ROS
- Install Git & Editor
<https://code.visualstudio.com/>
- Install NodeJS
<https://nodejs.org/en/download/>
- Install Sails

```
npm install sails -g
```

- If on Windows:

1. Install Ubuntu

- Got to Microsoft store
- Search for 'Ubuntu'
- Click get/install

2. Install ROS

- Follow ROS Kinetic installation + (this might take some time) <http://wiki.ros.org/kinetic/Installation/Ubuntu>

```
sudo apt-get install ros-kinetic-desktop-full
```

- And enviroment setup

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

- Test the installation running ROS

```
roscore
```

- Done for now

```
(ctrl+c)
```

1.2. Github

- Invite your personal Github account to acces Windesheim-Willy repos <https://github.com/orgs/Windesheim-Willy/people>
- Clone Git Repo

```
git clone https://github.com/Windesheim-Willy/repo-name
```

- Switch to test branch

```
git checkout -b origin/test
```

1.3. Compilation

Compilation is done via catkin, this is done to create a rospackage so that nodejs can run in a ROS environment.

```
cd WWEB/src
npm install
cd ..
source /opt/ros/kinetic/setup.bash
catkin_make
```

1.4. Testing/Debugging

Run without ros:

```
cd WWEB/src
sails lift (or node app.js)
```

Run with Ros:

1. Start Roscore

- Open a terminal (Ubuntu app on windows) -

```
cd WWEB
source devel/setup.bash
roscore
```

2. Run webplatform

- Open a terminal (Ubuntu app on windows) -

```
cd WWEB
source devel/setup.bash
roslaunch willyweb start.sh
```



The roslaunch command might not have access to port 80 for this to work use sudo -s

```
sudo -s
roslaunch willyweb start.sh
```

1.5. Running Scripts

In the same manner as you would do [Testing/Debugging](#) you can also run scripts. Scripts are located in the folder 'WWEB/src/scripts'.

1. Start Roscore

- Open a terminal (Ubuntu app on windows) -

```
cd WWEB
source devel/setup.bash
roscore
```

2. Run sending script

- Open a terminal (Ubuntu app on windows) -

```
cd WWEB
source devel/setup.bash
roslaunch willyweb scripts/send.js
```

3. Run receive script

- Open a terminal (Ubuntu app on windows) -

```
cd WWEB
source devel/setup.bash
roslaunch willyweb scripts/receive.js
```



Rosrun makes it possible to communicate with ROS because it is now run as a ROS package

The 'start.sh' script consist out of a simple run script which launches the webplatform



```
#!/usr/bin/env bash
node src/app.js
```