Advanced code juggling @MRT

Claudio Bandera

Or how the MRT tools can ease your everyday life

What are the MRT Tools?

Collection of command line tools, written in Python:

```
> mrt
```

What are they useful for?

- Working with a big codebase
- Interacting with gitlab
- Saving time on creating new code
- Automating reoccuring tasks

Which tools does it use under the hood?

- cmake
- catkin
- wstool
- gitlab-api
- docker
- doxygen
- rosbag

Finding help

```
> mrt pkg create --help

Usage: mrt pkg create [OPTIONS] PKG_NAME

  This is a package creation wizard, to help creating new catkin packages.
  You can specify whether to create a library or executable, ROS or non-ROS
  package and whether to create a Gitlab repo. Appropriate template files
  and directory tree are created. When creating the repo you can choose the
  namespace. The repo name is tested for conformity with the guidelines and
  conflicts with rosdep packages are avoided.

Options:
  -t [lib|exec]  Type: Choose between library or executable
  -r             Make ROS package
  -g             Create Git repository
  --help         Show this message and exit.
```

Finding help

```
> mrt pkg create --help

Usage: mrt pkg create [OPTIONS] PKG_NAME

  This is a package creation wizard, to help creating new catkin packages.
  You can specify whether to create a library or executable, ROS or non-ROS
  package and whether to create a Gitlab repo. Appropriate template files
  and directory tree are created. When creating the repo you can choose the
  namespace. The repo name is tested for conformity with the guidelines and
  conflicts with rosdep packages are avoided.

Options:
  -t [lib|exec]  Type: Choose between library or executable
  -r             Make ROS package
  -g             Create Git repository
  --help         Show this message and exit.
```

Autocompletion *(Now even for zsh!)*

```
> mrt pkg add sensor_[tab tab]
sensor_camera                     sensor-ibeo
sensor_camera_demo_tool           sensor_ibeo_demo_ros
sensor_camera_ros_tool            sensor_ibeo_feature_fusion_tool
sensor_ibeo                       sensor_ibeo_live_viewer_tool
...
```

Overview of commands

```
Commands:
  catkin       A wrapper for catkin.
  check        Test code in clean environment
  doc          Build and show the documentation of a package.
  gitlab       Gitlab related tools
  maintenance  Repair tools...
  pkg          Package related tasks...
  rosbag       A wrapper for rosbag.
  snapshot     Save or restore the current state of the...
  ws           A collection of tools to perform on a catkin...
  wstool       A wrapper for wstool.
```

Go out and explore!

Meanwhile there are about 47 nested commands (tendency rising)

Getting started

An easy exercise...

Creating a workspace

- All workspace relevant commands are nested under `mrt ws`

```
> mrt ws init my_first_ws
Creating workspace
Initializing catkin workspace in `/home/bandera/my_first_ws`.
...
> cd my_first_ws
```

Creating a workspace

- All workspace relevant commands are nested under `mrt ws`

```
> mrt ws init my_first_ws
Creating workspace
Initializing catkin workspace in `/home/bandera/my_first_ws`.
...
> cd my_first_ws
```

- Start by adding a package to your ws with `mrt pkg`

```
> mrt pkg add sensor_oxts_ros_tool
Search for package sensor_oxts_ros_tool
Found MRT/sensor_oxts_ros_tool
Cloning into '/home/bandera/my_first_ws/src/sensor_oxts_ros_tool'...
Resolving dependencies...
...
```

Creating a workspace

- All workspace relevant commands are nested under `mrt ws`

```
> mrt ws init my_first_ws
Creating workspace
Initializing catkin workspace in `/home/bandera/my_first_ws`.
...
> cd my_first_ws
```

- Start by adding a package to your ws with `mrt pkg`

```
> mrt pkg add sensor_oxts_ros_tool
Search for package sensor_oxts_ros_tool
Found MRT/sensor_oxts_ros_tool
Cloning into '/home/bandera/my_first_ws/src/sensor_oxts_ros_tool'...
Resolving dependencies...
...
```

- Inspect your ws

```
> mrt ws info
sensor_oxts                     git master  (-)    7b3dad5a5a5c gitlab.mrt.uni-karlsruhe.de/MRT/sensor_oxts.git
generic_logger                  git master  (-)    143fb9fd368d gitlab.mrt.uni-karlsruhe.de/MRT/generic_logger.
gnss_coordinate_transform       git master  (-)    56212d685f3f gitlab.mrt.uni-karlsruhe.de/MRT/gnss_coordinate
utils_ros                       git master  (-)    524bb2f34e18 gitlab.mrt.uni-karlsruhe.de/MRT/utils_ros.git
gnss_coordinate_transform_ros   git master  (-)    9fee0a9409b7 gitlab.mrt.uni-karlsruhe.de/MRT/gnss_coordinate
...
```

Build

- `mrt catkin` is a wrapper around `catkin` with additional options
    - `-rd` Check and resolve dependencies before building workspace.
    - `--eclipse` Create a eclipse project.
    - `--verbose` Compile in *very* verbose mode.
    - `--force-cmake` Force cmake to be run again (Helpful if new executables were created).
    - `-c` Continue with other packages upon failure.

Build

- `mrt catkin` is a wrapper around `catkin` with additional options
    - `-rd` Check and resolve dependencies before building workspace.
    - `--eclipse` Create a eclipse project.
    - `--verbose` Compile in *very* verbose mode.
    - `--force-cmake` Force cmake to be run again (Helpful if new executables were created).
    - `-c` Continue with other packages upon failure.
- Default settings will be set by mrt tools (Defaults can be altered with `mrt maintenance settings`)
    - Build in release mode (can be adjusted with `--debug`, `--release`)
    - Build with compiler warnings enabled (can be adjusted with `--warnings`, `--no-warnings`)

Build

- `mrt catkin` is a wrapper around `catkin` with additional options
    - `-rd` Check and resolve dependencies before building workspace.
    - `--eclipse` Create a eclipse project.
    - `--verbose` Compile in *very* verbose mode.
    - `--force-cmake` Force cmake to be run again (Helpful if new executables were created).
    - `-c` Continue with other packages upon failure.

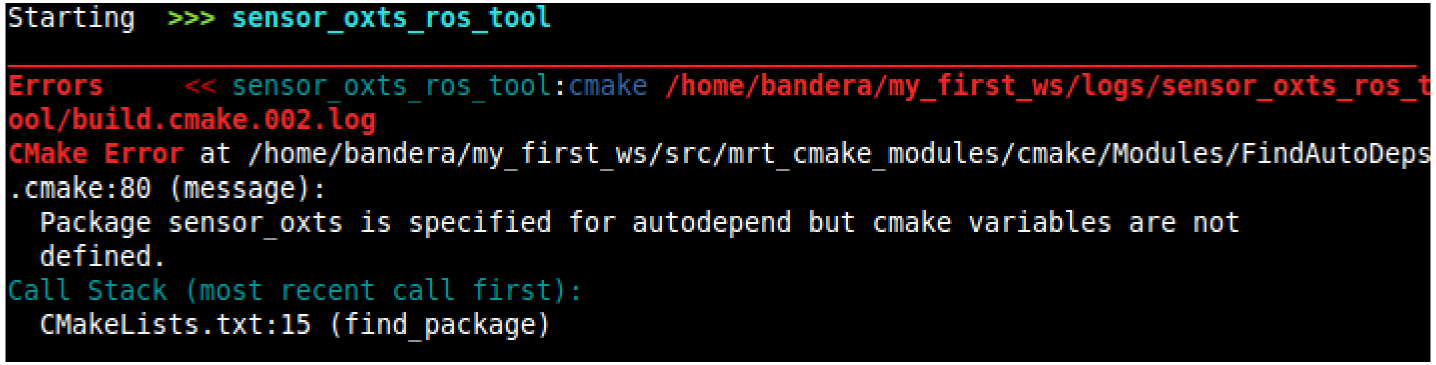
- Default settings will be set by mrt tools (Defaults can be altered with `mrt maintenance settings`)
    - Build in release mode (can be adjusted with `--debug`, `--release`)
    - Build with compiler warnings enabled (can be adjusted with `--warnings`, `--no-warnings`)
- Remove all compiled code
    - `mrt ws clean`

Build Errors

- If you get errors during compilation: **READ THEM**

Build Errors

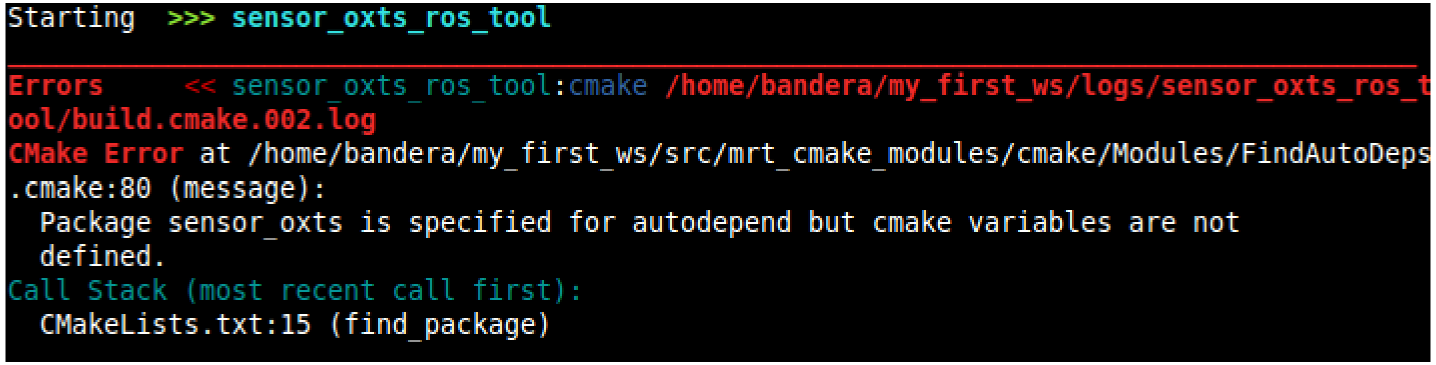
- If you get errors during compilation: **READ THEM**



```
Starting  >>> sensor_oxts_ros_tool
_______________________________________________________________________________
Errors     << sensor_oxts_ros_tool:cmake /home/bandera/my_first_ws/logs/sensor_oxts_ros_t
ool/build.cmake.002.log
CMake Error at /home/bandera/my_first_ws/src/mrt_cmake_modules/cmake/Modules/FindAutoDeps
.cmake:80 (message):
  Package sensor_oxts is specified for autodepend but cmake variables are not
  defined.
Call Stack (most recent call first):
  CMakeLists.txt:15 (find_package)
```

Build Errors

- If you get errors during compilation: **READ THEM**



```
Starting  >>> sensor_oxts_ros_tool
___________________________________________________________________________________
Errors     << sensor_oxts_ros_tool:cmake /home/bandera/my_first_ws/logs/sensor_oxts_ros_t
ool/build.cmake.002.log
CMake Error at /home/bandera/my_first_ws/src/mrt_cmake_modules/cmake/Modules/FindAutoDeps
.cmake:80 (message):
  Package sensor_oxts is specified for autodepend but cmake variables are not
  defined.
Call Stack (most recent call first):
  CMakeLists.txt:15 (find_package)
```

- In this case the dependency `sensor_oxts` is missing. Try any of these to install them:

```
> mrt ws resolve_deps
> mrt catkin build -rd
```

Going further

Create your own code

Create a package

- You can easily setup new packages

```
> mrt pkg create -g -r -t exec one_to_rule_them_all
Creating package with name.... one_to_rule_them_all_ros_tool
 --> Package type.... exec
 --> Create ROS Package.... YES
 --> Create  gitlab repository.... YES
 --> Package Maintainer.... Claudio Bandera <claudio.bandera@kit.edu>
```

Create a package

- You can easily setup new packages

```
> mrt pkg create -g -r -t exec one_to_rule_them_all
Creating package with name.... one_to_rule_them_all_ros_tool
 --> Package type.... exec
 --> Create ROS Package.... YES
 --> Create  gitlab repository.... YES
 --> Package Maintainer.... Claudio Bandera <claudio.bandera@kit.edu>
```

And what you get:

```
> tree .
├── CMakeLists.txt
├── launch
│   └── params
├── package.xml
├── README.md
├── res
├── src
└── test
.    └── test_one_to_rule_them_all_ros_tool.cpp
```

Creating a new ROS executable

```
> mrt pkg create_executable Ring_keeper
Do you need tf conversions? [y/N]: y
Do you need diagnostics? [y/N]: n
```

Creating a new ROS executable

```
> mrt pkg create_executable Ring_keeper
Do you need tf conversions? [y/N]: y
Do you need diagnostics? [y/N]: n
```

Using `mrt pkg create_executable` will:

- Add necessary dependencies to `package.xml`
- Add the following files:

```
.
├── src
│   └── ring_keeper
│       ├── ring_keeper.cpp               # <- Implementation Class
│       ├── ring_keeper.hpp               # <- Implementation Class Header
│       ├── ring_keeper_node.cpp          # <- Node Wrapper
│       └── ring_keeper_nodelet.cpp       # <- Nodelet Wrapper
├── launch
│   ├── params
│   │   └── ring_keeper_parameters.yaml # <- File for non-default params
│   ├── ring_keeper_node.launch           # <- Node launchfile
│   └── ring_keeper_nodelet.launch        # <- Nodelet launchfile
├── cfg
│   └── RingKeeper.mrtcfg                 # <- MRT Parameter file
└── nodelet_plugins.xml                # <- Declaration of nodelet plugins
```

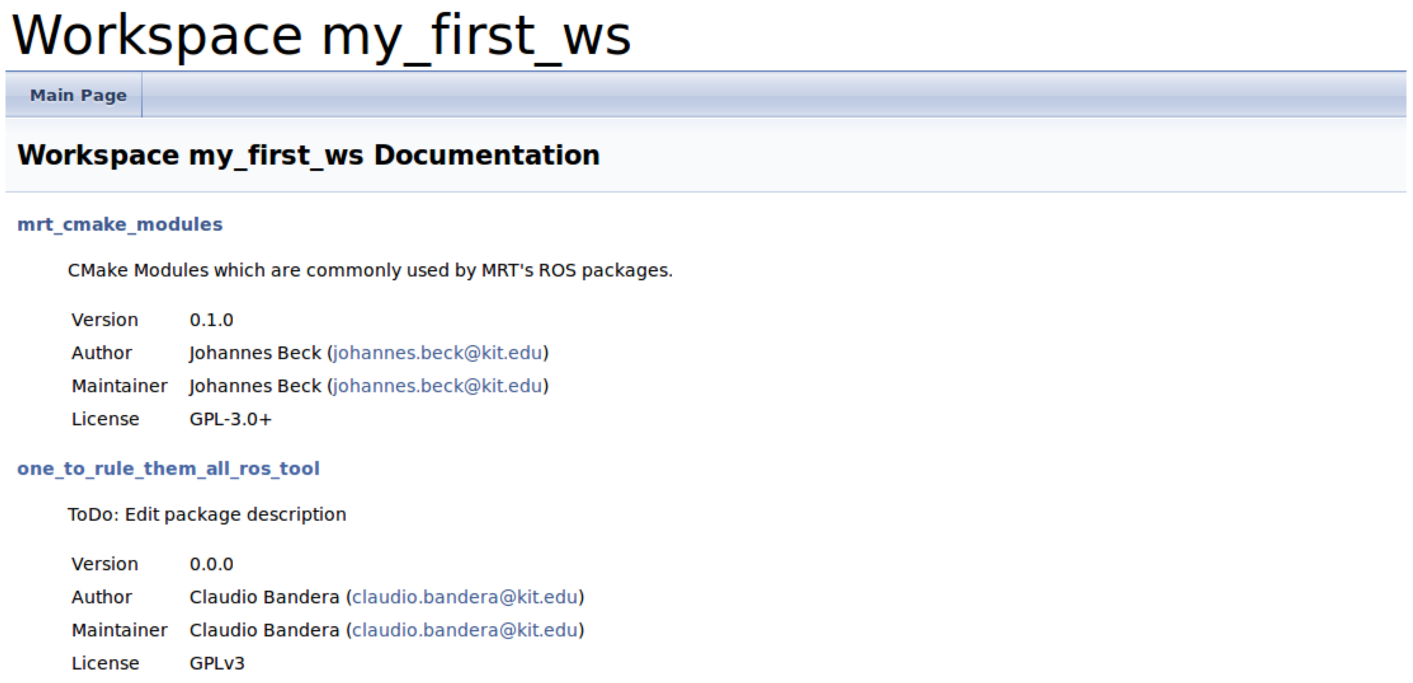
Create your documentation

```
> mrt doc build one_to_rule_them_all_ros_tool
> mrt doc show
```

Create your documentation

```
> mrt doc build one_to_rule_them_all_ros_tool
> mrt doc show
```

- This will compile all Doxygen comments in your code with default settings and display a workspace landing page.



Workspace my_first_ws

Workspace my_first_ws Documentation

mrt_cmake_modules

CMake Modules which are commonly used by MRT's ROS packages.

Version	0.1.0
Author	Johannes Beck (johannes.beck@kit.edu)
Maintainer	Johannes Beck (johannes.beck@kit.edu)
License	GPL-3.0+

one_to_rule_them_all_ros_tool

ToDo: Edit package description

Version	0.0.0
Author	Claudio Bandera (claudio.bandera@kit.edu)
Maintainer	Claudio Bandera (claudio.bandera@kit.edu)
License	GPLv3

Test your code

- Write tests and run them! *(There is already a demo test file in your pkg)*

```
> mrt catkin run_tests
```

Test your code

- Write tests and run them! *(There is already a demo test file in your pkg)*

```
> mrt catkin run_tests
```

- Test whether it will compile in a clean environment:

```
> mrt check pkg one_to_rule_them_all_ros_tool
```

Test your code

- Write tests and run them! *(There is already a demo test file in your pkg)*

```
> mrt catkin run_tests
```

- Test whether it will compile in a clean environment:

```
> mrt check pkg one_to_rule_them_all_ros_tool
```

  - This will ssh into the mrtknecht,
  - start up a fresh docker container with a fresh & clean install
  - create a new workspace
  - clone your package
  - resolve dependencies
  - and compile it.

Collaborate

- Easily add students to your repositories

```
> mrt gitlab permissions add_user
```

Preserve Data

Or how to find those rosbags again

Record Rosbags

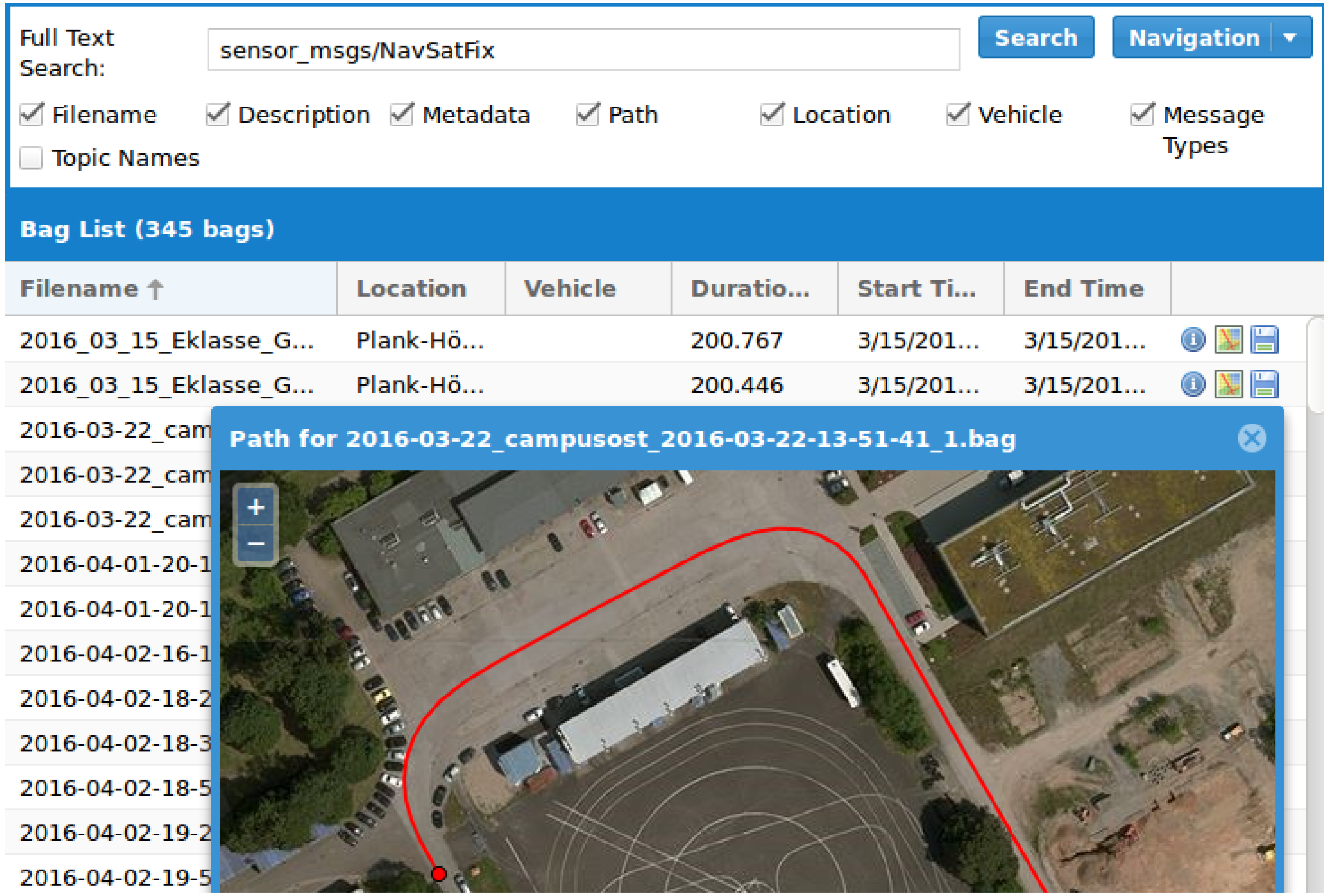
- Use the mrt wrapper for recording rosbags:

```
> mrt rosbag record -o MyFirstRosbag /chatter
```

- This will prompt your for more information about
  - Vehicle
  - Location
  - Sensors
  - Description
- This metadata is stored inside the bagfile

Find Rosbags

- Go to http://rosbag/ to search existing recordings



Full Text Search: sensor_msgs/NavSatFix

☑ Filename ☑ Description ☑ Metadata ☑ Path ☑ Location ☑ Vehicle ☑ Message Types
☐ Topic Names

Bag List (345 bags)

Filename ↑	Location	Vehicle	Duratio...	Start Ti...	End Time
2016_03_15_Eklasse_G...	Plank-Hö...		200.767	3/15/201...	3/15/201...
2016_03_15_Eklasse_G...	Plank-Hö...		200.446	3/15/201...	3/15/201...
2016-03-22_cam					
2016-03-22_cam					
2016-03-22_cam					
2016-04-01-20-1					
2016-04-01-20-1					
2016-04-02-16-1					
2016-04-02-18-2					
2016-04-02-18-3					
2016-04-02-18-5					
2016-04-02-19-2					
2016-04-02-19-5					

Path for 2016-03-22_campusost_2016-03-22-13-51-41_1.bag

Save Demos

- Make Snapshots of succesfull demos

```
> mrt snapshot create MyFirstDemo
Wrote snapshot to /home/bandera/my_first_ws/MyFirstDemo_160721.snapshot
```

Save Demos

- Make Snapshots of succesfull demos

```
> mrt snapshot create MyFirstDemo
Wrote snapshot to /home/bandera/my_first_ws/MyFirstDemo_160721.snapshot
```

- So they can be reproduced

```
> mrt snapshot restore MyFirstDemo_160721.snapshot
...
```

Housekeeping

Or how to handle ALL those repos

Handling many packages at once

- Thankfully, there exist some tools to get an insight of whats going on:
    - See branch, unpushed commits, modified [and untracked] files

      ```
      mrt ws info [-u]
      ```

Handling many packages at once

- Thankfully, there exist some tools to get an insight of whats going on:
  - See branch, unpushed commits, modified [and untracked] files

    ```
    mrt ws info [-u]
    ```

  - Perform a `git status` in every repo.

    ```
    mrt ws status
    ```

Handling many packages at once

- Thankfully, there exist some tools to get an insight of whats going on:
  - See branch, unpushed commits, modified [and untracked] files

    ```
    mrt ws info [-u]
    ```

  - Perform a `git status` in every repo.

    ```
    mrt ws status
    ```

  - Perform a `git pull` in every repo.

    ```
    mrt ws update
    ```

Handling many packages at once

- Thankfully, there exist some tools to get an insight of whats going on:
    - See branch, unpushed commits, modified [and untracked] files

      ```
      mrt ws info [-u]
      ```

    - Perform a `git status` in every repo.

      ```
      mrt ws status
      ```

    - Perform a `git pull` in every repo.

      ```
      mrt ws update
      ```

    - Perform anything in every repo

      ```
      mrt wstool foreach 'touch claudio_was_here.txt'
      ```

Inspect dependencies #1

Long dependency chains can get confusing at times, try these:

Inspect dependencies #1

Long dependency chains can get confusing at times, try these:

- List dependencies

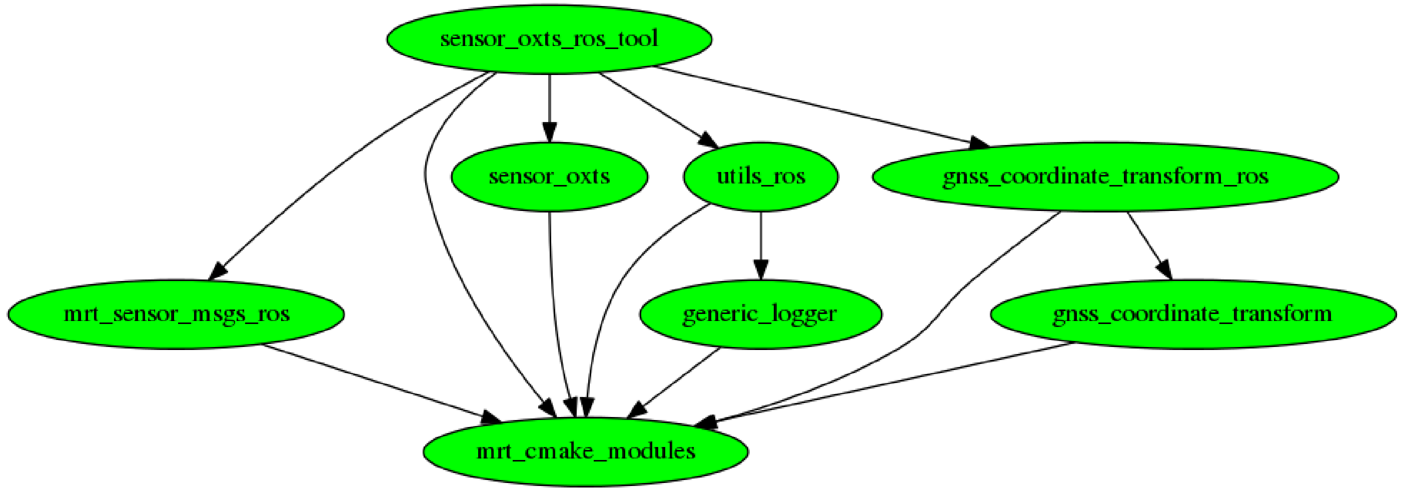
```
> mrt pkg deps show sensor_oxts_ros_tool
Gitlab dependencies
===================
gnss_coordinate_transform
utils_ros
sensor_oxts
generic_logger
mrt_sensor_msgs_ros
gnss_coordinate_transform_ros
mrt_cmake_modules
_
Apt-get dependencies
====================
message_runtime
catkin
roscpp
std_msgs
roslib
sensor_msgs
eigen_conversions
diagnostic_updater
gtest
geometry_msgs
...
```

Inspect dependencies #2

Long dependency chains can get confusing at times, try these:

- Draw dependency tree

```
> mrt pkg deps draw sensor_oxts_ros_tool --repos-only
```



Inspect dependencies #3

Long dependency chains can get confusing at times, try these:

- Lookup reverse dependencies

```
> mrt pkg deps rlookup sensor_oxts_ros_tool
I found the following packages relying on sensor_oxts_ros_tool:
-
MRT/vehicle_config_files:
- On branch 'demo_follow_ibeo_object': [u'depend']
- On branch 'gcdc': [u'depend']
- On branch 'devel_sahin': [u'depend']
MRT/control_reference_ros_tool:
- On branch 'master': [u'build_depend', u'exec_depend']
gcdc/gpsd_ros_tool:
- On branch 'master': [u'depend']
```

Cleaning up

- Try to keep your workspaces small (if possible)

Cleaning up

- Try to keep your workspaces small (if possible)
- Remove packages (safely) if not needed.

```
> mrt pkg remove sensor_oxts_ros_tool
You have the following uncommited changes:
sensor_oxts_ros_tool
M       sensor_oxts_ros_tool/README.md
Are you sure you want to continue? [y/N]:
```

Cleaning up

- Try to keep your workspaces small (if possible)
- Remove packages (safely) if not needed.

```
> mrt pkg remove sensor_oxts_ros_tool
You have the following uncommited changes:
sensor_oxts_ros_tool
M       sensor_oxts_ros_tool/README.md
Are you sure you want to continue? [y/N]:
```

- Try to find packages that are no longer needed:

```
> mrt ws tidy
Package sensor_oxts_ros_tool has no parents. Delete? [y/N]: y
Removing sensor_oxts_ros_tool
```

Maintenance

- All persistent settings and cache files lie in `~/.mrtgitlab`
- Check out the configuration options you have:

```
> mrt maintenance settings
```

Maintenance

- All persistent settings and cache files lie in `~/.mrtgitlab`
- Check out the configuration options you have:

  ```
  > mrt maintenance settings
  ```

- Delete or change your saved credentials

  ```
  > mrt maintenance credentials
  ```

Maintenance

- All persistent settings and cache files lie in `~/.mrtgitlab`
- Check out the configuration options you have:

```
> mrt maintenance settings
```

- Delete or change your saved credentials

```
> mrt maintenance credentials
```

- Find more (experimental) commands under "maintenance":
  - credentials
  - rename_pkg
  - settings
  - update_cached_deps
  - update_cmakelists
  - update_repo_cache
  - update_rosinstall
  - update_url_in_package_xml

Behind the scenes

Or how you can help yourself

The codebase

- All code located in `mrt_build` repo.
- Python module `mrt_tools` contains all code
    - Dependencies are listed in requirements.txt
    - Use `setup.py` to install
    - Use `tox` to run tests
- Tools get installed into own virtualenv, see documentation in repo.

Adding new commands

- Library is written with the help of click
- To add a custom command:
    - create a new file starting with `mrt_` in the commands folder.
    - Use decorator to define new command:

```
@click.command()
@click.option('--count', default=1, help='Number of greetings.')
@click.option('--name', prompt='Your name',
              help='The person to greet.')
def hello(count, name):
    """Simple program that greets NAME for a total of COUNT times."""
    for x in range(count):
        click.echo('Hello %s!' % name)
```

Credentials

- User credentials can be stored in several different places:
    - Gnome Keyring
    - Encoded File
    - Only in memory
- Additionally, git credentials can be passed to `git-credentials-cache`

```
>mrt maintenance credentials show
Gitlab credentials
==================
(Current setting: 'GnomeCredentialManager')
Username: bandera
Password: ******
Token   : xxxxxxxxxxxxxxxxxxxxxx
-
Git credentials
==================
user.name    : Claudio Bandera
user.email   : claudio.bandera@kit.edu
cached creds.: Yes
```

\`\`\`