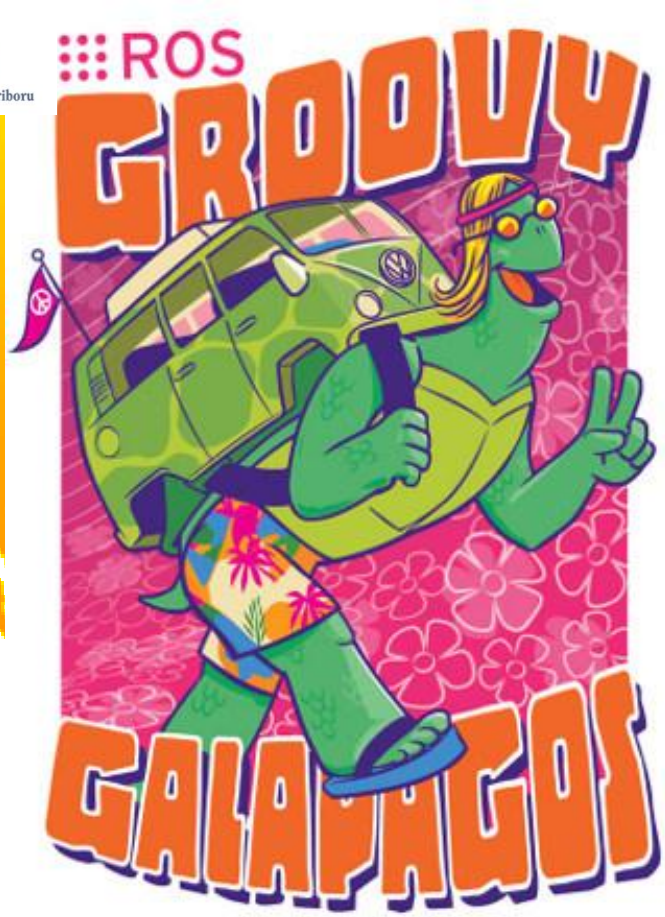


ROS tutorial

ROS (Robot Operating System) is an open-source, meta-operating system for robots.

Peter Lepej
September 2013
UNI-MB FERI

Naložba v vašo prihodnost
Operacijo delno financira Evropska unija
Evropski sklad za regionalni razvoj

Investition in Ihre Zukunft
Operation teilfinanziert von der Europäischen Union
Europäischer Fonds für regionale Entwicklung

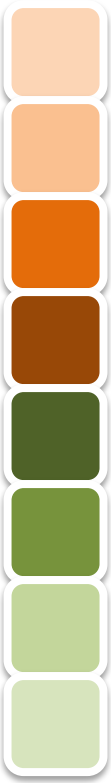
REPUBLIKA SLOVENIJA
MINISTRSTVO ZA GOSPODARSKI
RAZVOJ IN TEHNOLOGIJO

ROS TUTORIAL WORK PLAN

- Introduction to ROS
- Online tutorial step-by-step:
  - ROS Basics
  - ROS Topics and Messages
  - ROS C++ Example
  - ROS Services and Parameters
  - ROS C++ Example
  - ROS Tools
- ROS Cheat Sheet.
- Work and Learn.
- At the end build groups and assign projects which will be presented at the end of summer school.
- If you have any question do not hesitate to ask!

Naložba v vašo prihodnost
Operacijo delno financira Evropska unija
Evropski sklad za regionalni razvoj
Investition in Ihre Zukunft
Operation teilfinanziert von der Europäischen Union
Europäischer Fonds für regionale Entwicklung
REPUBLIKA SLOVENIJA
MINISTRSTVO ZA GOSPODARSKI
RAZVOJ IN TEHNOLOGIJO

CONTENT



PRESENTATION

BASIC

NODES

TOPICS and MESSAGES

ROS and C++ (Simple Publisher and Subscriber)

SERVICES and PARAMETERS

ROS and C++ (Simple Service and Client)

TOOLS

INTRODUCTION TO ROS

- *ROS (Robot Operating System) provides libraries and tools to help software developers create robot applications. It provides hardware abstraction, device drivers, libraries, visualizers, message-passing, package management, and more. ROS is licensed under an open source, BSD license*.

- It enables quick and easy start in field of mobile robotics.

- ROS current distribution: Groovy Galapagos

- ROS Documentation: http://wiki.ros.org/
- ROS Tutorials: http://wiki.ros. org/ROS/Tutorials
  - Where the following material is taken from.

Naložba v vašo prihodnost
Operacijo delno financira Evropska unija
Evropski sklad za regionalni razvoj
Investition in Ihre Zukunft
Operation teilfinanziert von der Europäischen Union
Europäischer Fonds für regionale Entwicklung
REPUBLIKA SLOVENIJA
MINISTRSTVO ZA GOSPODARSKI
RAZVOJ IN TEHNOLOGIJO

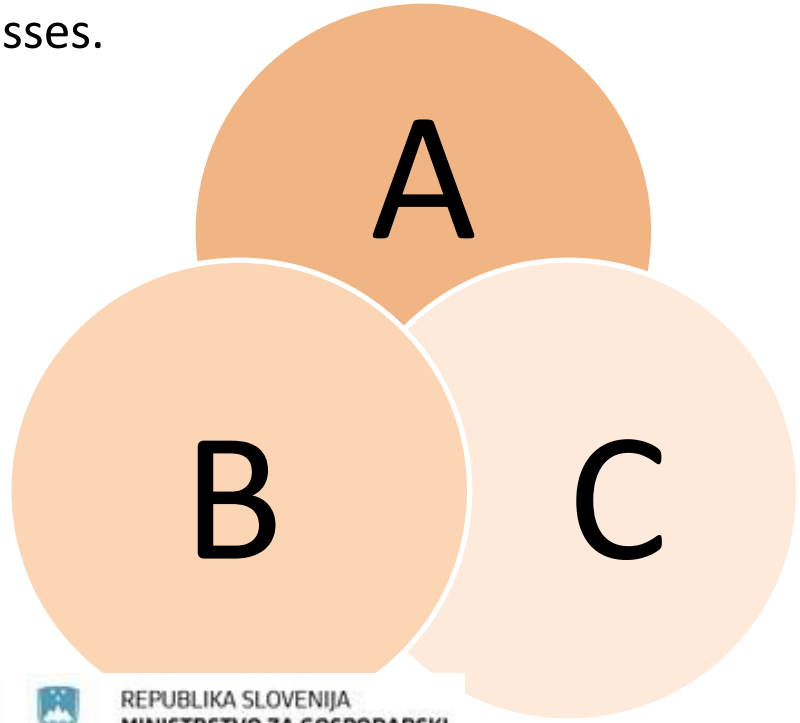
INTRODUCTION TO ROS

- ROS is to support code reuse in robotics research and development.
- *ROS is a distributed framework of processes (Nodes) .*
- Processes can be grouped into Packages and Stacks.
- Philosophy: ROS libraries should have clean functional interfaces.
- Language independence: Python and C++ (and others).
- All ROS core code is licensed BSD, so it is easy to integrate in your project.

Naložba v vašo prihodnost
Operacijo delno financira Evropska unija
Evropski sklad za regionalni razvoj
Investition in Ihre Zukunft
Operation teilfinanziert von der Europäischen Union
Europäischer Fonds für regionale Entwicklung
REPUBLIKA SLOVENIJA
MINISTRSTVO ZA GOSPODARSKI
RAZVOJ IN TEHNOLOGIJO

Concepts

- ROS has three levels of concepts:
  - A) **Filesystem level**: ROS resources on disk.
  - B) **Computation Graph level:**
    Peer-to-peer network of ROS processes.
  - C) **Community level:**
    For everybody!



Filesystem Level

The filesystem level that you encounter on disk, such as:

- **PACKAGES:** Packages are the main unit for organizing software in ROS, e.g. ROS runtime processes (nodes), ROS-dependent library, datasets, configuration files.
- **MANIFEST:** Manifests (manifest.xml) provide metadata about a package (e.g. dependencies, compiler flags).
- **STACKS:** Stacks are collections of packages that provide aggregate functionality, such as a navigation stack.
- **STACK MANIFEST:** Stack manifests (stack.xml) provide data about a stack (e.g. dependencies on other stacks).

Naložba v vašo prihodnost
Operacijo delno financira Evropska unija
Evropski sklad za regionalni razvoj
Investition in Ihre Zukunft
Operation teilfinanziert von der Europäischen Union
Europäischer Fonds für regionale Entwicklung
REPUBLIKA SLOVENIJA
MINISTRSTVO ZA GOSPODARSKI
RAZVOJ IN TEHNOLOGIJO

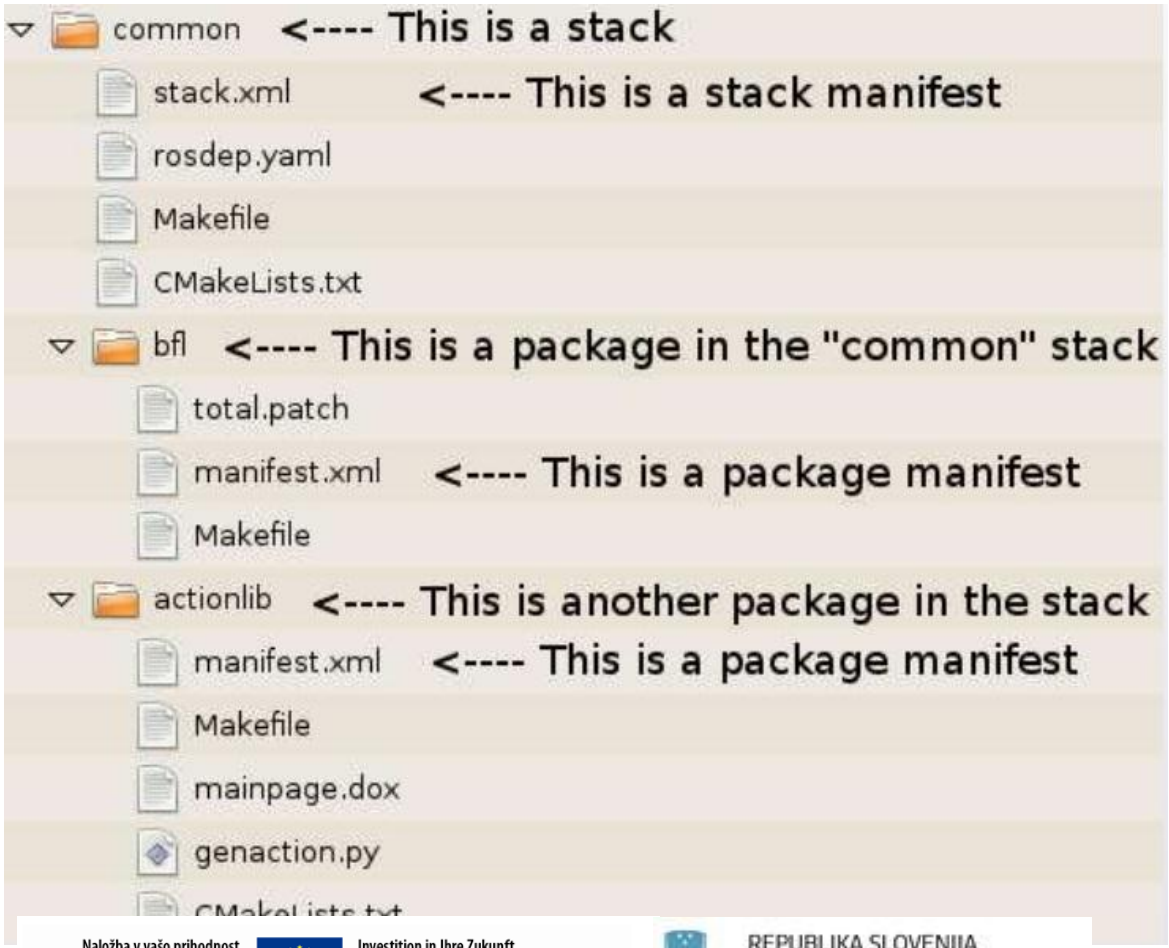
Filesystem Level

- **MESSAGE (msg) types:** Message descriptions, stored in my_package/msg/MyMessageType.msg, define the data structures for messages sent in ROS.
- **SERVICE (srv) types:** Service descriptions, stored in my_package/srv/MyServiceType.srv, define the request and response data structures for services in ROS.

Naložba v vašo prihodnost
Operacijo delno financira Evropska unija
Evropski sklad za regionalni razvoj
Investition in Ihre Zukunft
Operation teilfinanziert von der Europäischen Union
Europäischer Fonds für regionale Entwicklung
REPUBLIKA SLOVENIJA
MINISTRSTVO ZA GOSPODARSKI
RAZVOJ IN TEHNOLOGIJO

Filesystem Structure

(disk location:/opt/ros/groovy/common)



Naložba v vašo prihodnost
Operacijo delno financira Evropska unija
Evropski sklad za regionalni razvoj
Investition in Ihre Zukunft
Operation teilfinanziert von der Europäischen Union
Europäischer Fonds für regionale Entwicklung
REPUBLIKA SLOVENIJA
MINISTRSTVO ZA GOSPODARSKI
RAZVOJ IN TEHNOLOGIJO

Distributer Compmutation Level

Computation Graph: Peer-to-peer network of ROS processes that are processing data together. The basic Computation Graph concepts of ROS are:

- Nodes
- Master (roscore)
- Parameter server
- Message
- Topics
- Services
- Bags

Let‘s go in details…

Naložba v vašo prihodnost
Operacijo delno financira Evropska unija
Evropski sklad za regionalni razvoj
Investition in Ihre Zukunft
Operation teilfinanziert von der Europäischen Union
Europäischer Fonds für regionale Entwicklung
REPUBLIKA SLOVENIJA
MINISTRSTVO ZA GOSPODARSKI
RAZVOJ IN TEHNOLOGIJO

Computation Graph

NODES

- NODES are processes that PERFORM COMPUTATION.
- ROS is designed to be modular, a robot control system will usually comprise many nodes.
- For example, one node controls a laser range-finder, another node performs localization.
- A ROS node is written with the use of a ROS client library, such as roscpp or rospy.

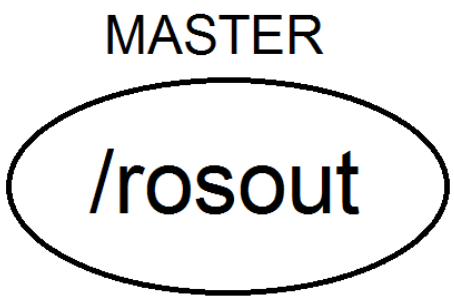
Naložba v vašo prihodnost
Operacijo delno financira Evropska unija
Evropski sklad za regionalni razvoj
Investition in Ihre Zukunft
Operation teilfinanziert von der Europäischen Union
Europäischer Fonds für regionale Entwicklung
REPUBLIKA SLOVENIJA
MINISTRSTVO ZA GOSPODARSKI RAZVOJ IN TEHNOLOGIJO

Computation Graph

MASTER

- Provides name registration and lookup to the rest of the Computation Graph.
- Without the Master, nodes would not be able to find each other, exchange messages, or invoke services.
- The master stores topics and services registration information for ROS nodes.

```
$roscore
```



Naložba v vašo prihodnost
Operacijo delno financira Evropska unija
Evropski sklad za regionalni razvoj
Investition in Ihre Zukunft
Operation teilfinanziert von der Europäischen Union
Europäischer Fonds für regionale Entwicklung
REPUBLIKA SLOVENIJA
MINISTRSTVO ZA GOSPODARSKI
RAZVOJ IN TEHNOLOGIJO

Computation Graph

PARAMETER SERVER

- The Parameter Server allows data to be stored by key in a central location.
- It is part of the Master.
- They are global variables.
- Part of ROS MASTER.
- Data type:
  - 32-bit integers
  - booleans
  - strings
  - doubles,...

```
$rosparam
```

Naložba v vašo prihodnost
Operacijo delno financira Evropska unija
Evropski sklad za regionalni razvoj
Investition in Ihre Zukunft
Operation teilfinanziert von der Europäischen Union
Europäischer Fonds für regionale Entwicklung
REPUBLIKA SLOVENIJA
MINISTRSTVO ZA GOSPODARSKI
RAZVOJ IN TEHNOLOGIJO

Computation Graph

MESSAGES

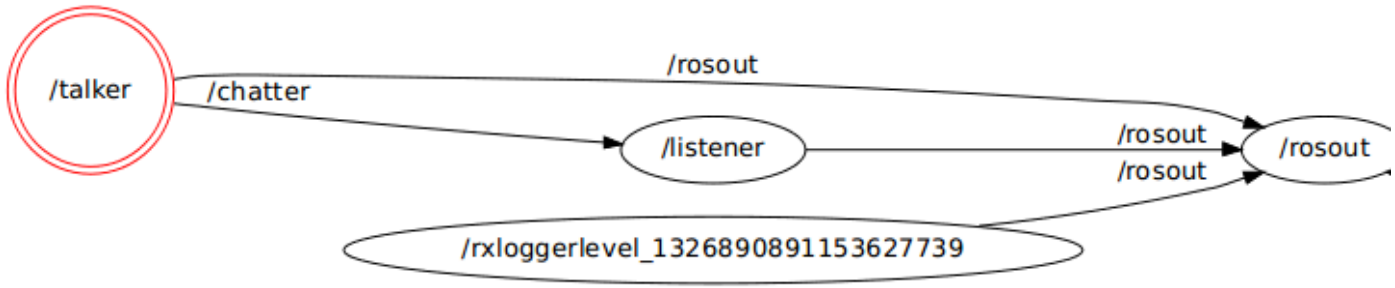
- MESSAGES: Nodes communicate with each other by passing messages.
- A message is simply a data structure of typed fields.
- Standard primitive types (integer, floating point, boolean, etc.) are supported, as are arrays of primitive types.
- Messages can include arbitrarily nested structures and arrays (much like C structs).

Naložba v vašo prihodnost
Operacijo delno financira Evropska unija
Evropski sklad za regionalni razvoj
Investition in Ihre Zukunft
Operation teilfinanziert von der Europäischen Union
Europäischer Fonds für regionale Entwicklung
REPUBLIKA SLOVENIJA
MINISTRSTVO ZA GOSPODARSKI
RAZVOJ IN TEHNOLOGIJO

Computation Graph

TOPICS

- A node sends out a message by publishing it to a given topic.
- The topic is a name that is used to identify the content of the message.
- A node that is interested in a certain kind of data will subscribe to the appropriate topic.
- In general, publishers and subscribers are not aware of each others existence (decoupling).
- Logically, one can think of a topic as a strongly typed message bus. Each bus has a name, and anyone can connect to the bus to send or receive messages as long as they are the right type.



Naložba v vašo prihodnost
Operacijo delno financira Evropska unija
Evropski sklad za regionalni razvoj
Investition in Ihre Zukunft
Operation teilfinanziert von der Europäischen Union
Europäischer Fonds für regionale Entwicklung
REPUBLIKA SLOVENIJA
MINISTRSTVO ZA GOSPODARSKI
RAZVOJ IN TEHNOLOGIJO

Computation Graph

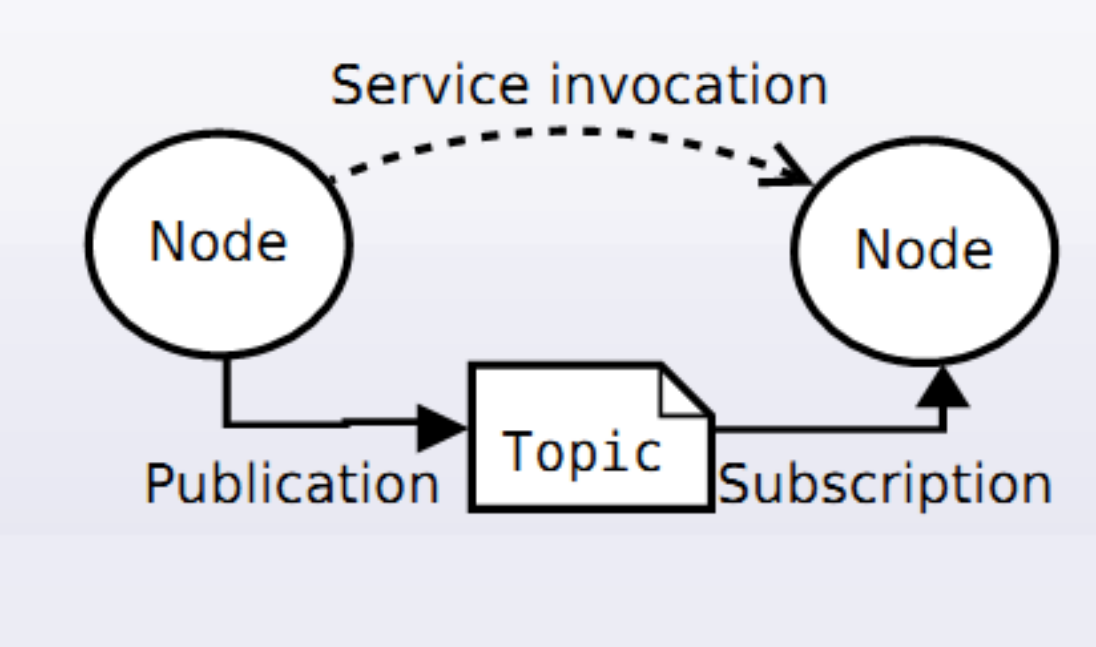
SERVICES

- Publish / subscribe model: many-to-many (messages)
- Request / reply: services
- Pair of message structures: one for the request and one for the reply.
- **A providing node offers a service under a name and a client uses the service by sending the request message and awaiting the reply.**

Naložba v vašo prihodnost
Operacijo delno financira Evropska unija
Evropski sklad za regionalni razvoj
Investition in Ihre Zukunft
Operation teilfinanziert von der Europäischen Union
Europäischer Fonds für regionale Entwicklung
REPUBLIKA SLOVENIJA
MINISTRSTVO ZA GOSPODARSKI
RAZVOJ IN TEHNOLOGIJO

Computation Graph

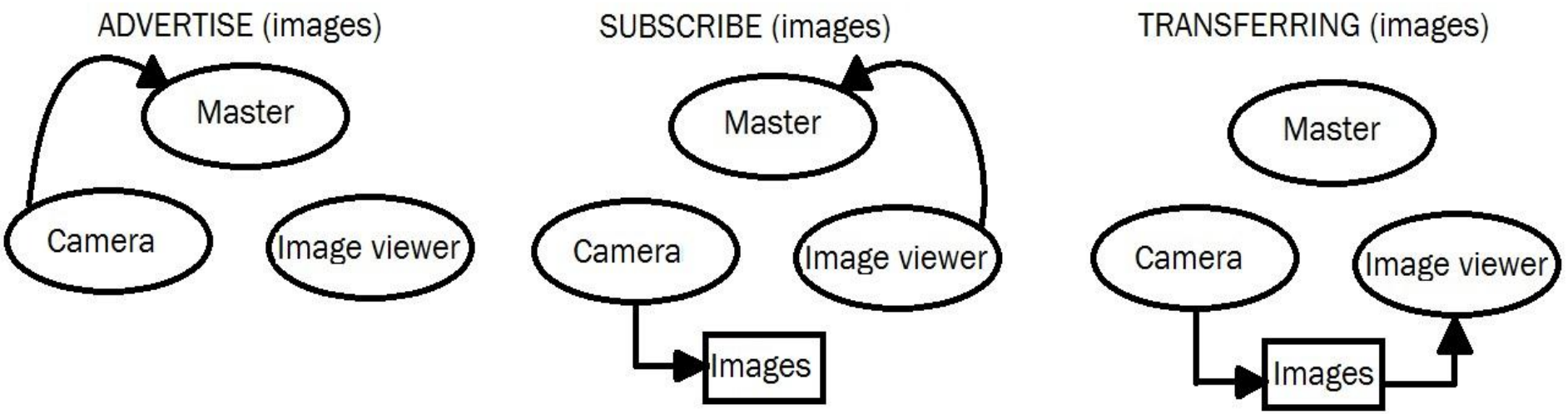
MESSAGE vs. SERVICES



Naložba v vašo prihodnost
Operacijo delno financira Evropska unija
Evropski sklad za regionalni razvoj
Investition in Ihre Zukunft
Operation teilfinanziert von der Europäischen Union
Europäischer Fonds für regionale Entwicklung
REPUBLIKA SLOVENIJA
MINISTRSTVO ZA GOSPODARSKI
RAZVOJ IN TEHNOLOGIJO

Computation Graph

MESSAGE COMUNICATION EXAMPLE



Naložba v vašo prihodnost
Operacijo delno financira Evropska unija
Evropski sklad za regionalni razvoj
Investition in Ihre Zukunft
Operation teilfinanziert von der Europäischen Union
Europäischer Fonds für regionale Entwicklung
REPUBLIKA SLOVENIJA
MINISTRSTVO ZA GOSPODARSKI
RAZVOJ IN TEHNOLOGIJO

ROS Community Level

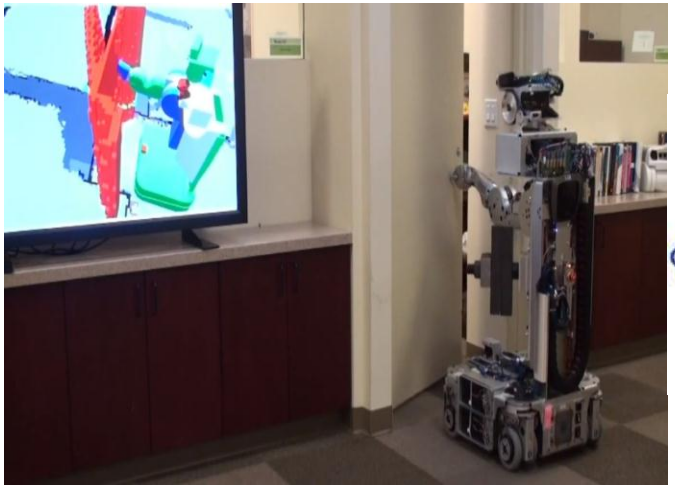
- **Enable separate communities to exchange software and knowledge. Resources:**
  - Distributions: ROS Distributions are collections of versioned stacks that you can install. (Comparable to Linux distributions).
  - Repositories: Different institutions can develop and release their own robot software components.
  - The ROS Wiki: The ROS community Wiki is the main forum for documenting information about ROS.
  - Blog

http://wiki.ros.org/

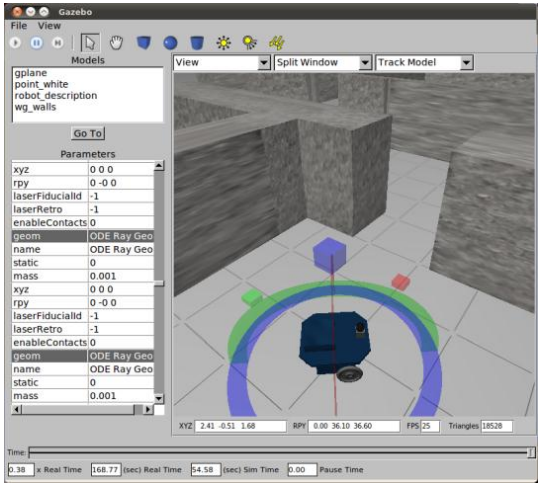
Naložba v vašo prihodnost
Operacijo delno financira Evropska unija
Evropski sklad za regionalni razvoj
Investition in Ihre Zukunft
Operation teilfinanziert von der Europäischen Union
Europäischer Fonds für regionale Entwicklung
REPUBLIKA SLOVENIJA
MINISTRSTVO ZA GOSPODARSKI RAZVOJ IN TEHNOLOGIJO

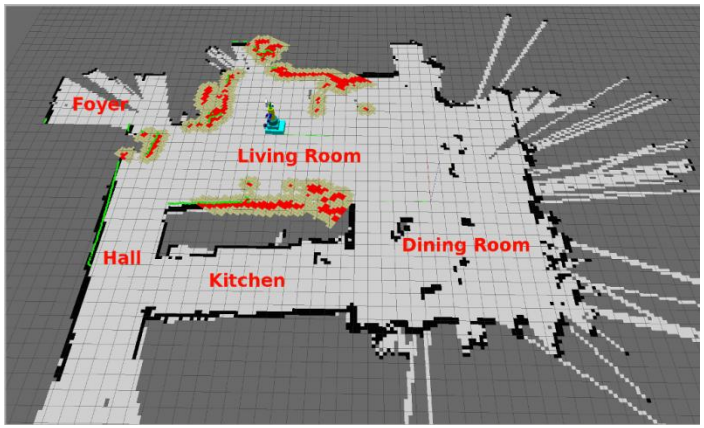
ROS Usage

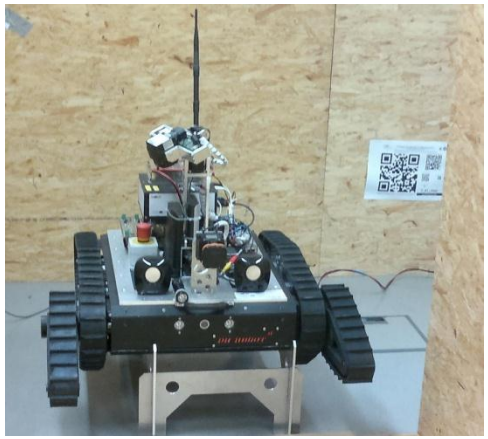
WHAT CAN WE DO WITH ROS?



ROS industrial™







Naložba v vašo prihodnost
Operacijo delno financira Evropska unija
Evropski sklad za regionalni razvoj

Investition in Ihre Zukunft
Operation teilfinanziert von der Europäischen Union
Europäischer Fonds für regionale Entwicklung

REPUBLIKA SLOVENIJA
MINISTRSTVO ZA GOSPODARSKI
RAZVOJ IN TEHNOLOGIJO

ROS CONTENT

PRESENTATION

BASIC

NODES

TOPICS and MESSAGES

ROS and C++ (Simple Publisher and Subscriber)

SERVICES and PARAMETERS

ROS and C++ (Simple Service and Client)

TOOLS

Naložba v vašo prihodnost
Operacijo delno financira Evropska unija
Evropski sklad za regionalni razvoj
Investition in Ihre Zukunft
Operation teilfinanziert von der Europäischen Union
Europäischer Fonds für regionale Entwicklung
REPUBLIKA SLOVENIJA
MINISTRSTVO ZA GOSPODARSKI
RAZVOJ IN TEHNOLOGIJO

Tutorial

INSTALING AND CONFIGURING ROS ENVIROMENT

- **Let‘s check our environment setup:**

```
$export | grep ROS
```

//ROS distribution, directory, master uri, package path and other… is shown.

- **Two available methods for organizing and building ROS code:**
  - **1. catkin:** standard cmake conversions and more sophisticated
  - **2. rosbuild:** easy to use and simple

- **To get access to ROS commands we need to setup source for ROS, we do this in .bashrc , so we don't have to run it every time:**

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

//You might done this step in installation tutorial.

Naložba v vašo prihodnost
Operacijo delno financira Evropska unija
Evropski sklad za regionalni razvoj
Investition in Ihre Zukunft
Operation teilfinanziert von der Europäischen Union
Europäischer Fonds für regionale Entwicklung
REPUBLIKA SLOVENIJA
MINISTRSTVO ZA GOSPODARSKI RAZVOJ IN TEHNOLOGIJO

Tutorial

INSTALING AND CONFIGURING ROS ENVIROMENT

- Create our workspace:
  - Install rosws:

```
$sudo apt-get install python-rosinstall
```

  - Create new workspace which extends set of packages installed in /opt/ros/groovy:

```
$rosws init ~/rosw /opt/ros/groovy
```

  - **Add source to our folder:**

```
$echo "source ~/rosw/setup.bash" >> ~/.bashrc
```

    //Restart terminal.

  - Create new sandbox directory which will be used by rosws:

```
$mkdir ~/rosw/sandbox
$rosws set ~/rosw/sandbox  //add new element (yes)
```

  - Check you $ROS_PACKAGE_PATH:

```
$export | grep ROS
```

    // It should be set like:
    /home/your_user_name/rosw/sandbox:/opt/ros/groovy/share:/opt/ros/groovy/stacks

Naložba v vašo prihodnost
Operacijo delno financira Evropska unija
Evropski sklad za regionalni razvoj
Investition in Ihre Zukunft
Operation teilfinanziert von der Europäischen Union
Europäischer Fonds für regionale Entwicklung
REPUBLIKA SLOVENIJA
MINISTRSTVO ZA GOSPODARSKI
RAZVOJ IN TEHNOLOGIJO

BASICS

Navigating the ROS filesystem

- Tools for easier work with a big numer of files and packages.
- ROS tools are working only in $ROS_PACKAGE_PATH directory.
- Command structure:

```
$command file_command name_file parameter1 parameter2…
```

- For each command exist help, who also works whit subcommands:

```
$command –h
$command subcommand -h
```

Naložba v vašo prihodnost
Operacijo delno financira Evropska unija
Evropski sklad za regionalni razvoj
Investition in Ihre Zukunft
Operation teilfinanziert von der Europäischen Union
Europäischer Fonds für regionale Entwicklung
REPUBLIKA SLOVENIJA
MINISTRSTVO ZA GOSPODARSKI
RAZVOJ IN TEHNOLOGIJO

BASICS

Using rospack and rosstack

- ROS command rospack and rosstack allow you to get information about packages and stacks.

- Usage:

```
$rospack find [package_name]
$rosstack find [stack_name]
```

- Let us try with:

```
$rospack find roscpp
```

```
YOUR_INSTALL_PATH/share/roscpp
```

Naložba v vašo prihodnost
Operacijo delno financira Evropska unija
Evropski sklad za regionalni razvoj
Investition in Ihre Zukunft
Operation teilfinanziert von der Europäischen Union
Europäischer Fonds für regionale Entwicklung
REPUBLIKA SLOVENIJA
MINISTRSTVO ZA GOSPODARSKI
RAZVOJ IN TEHNOLOGIJO

BASICS

Using roscd

- Command roscd (Change Directory - change the current working directory to a specific Folder). It allows you to change directory directly to a package or a stack.
- Usage:

```
$roscd [locationname[/subdir]]
```

- Now run:

```
$roscd roscpp
```

- To verify that we have changed to the roscpp package directory. Now let's print the working directory using the command **pwd** (Print Working Directory).

```
$pwd
```

- You should see:

```
YOUR_INSTALL_PATH/share/roscpp
```

Naložba v vašo prihodnost
Operacijo delno financira Evropska unija
Evropski sklad za regionalni razvoj
Investition in Ihre Zukunft
Operation teilfinanziert von der Europäischen Union
Europäischer Fonds für regionale Entwicklung
REPUBLIKA SLOVENIJA
MINISTRSTVO ZA GOSPODARSKI
RAZVOJ IN TEHNOLOGIJO

BASICS

Using roscd

- Command roscd can also move to a subdirectory of a package or stack.

- Try:

```
$roscd roscpp/cmake
```

- And again:

```
$pwd    //print working directory
```

- You should see:

```
YOUR_INSTALL_PATH/share/roscpp
```

- Command roscd log will take you to the folder where ROS stores log files. Note that if you have not run any ROS programs yet, this will yield an error saying that it does not yet exist.

Naložba v vašo prihodnost
Operacijo delno financira Evropska unija
Evropski sklad za regionalni razvoj
Investition in Ihre Zukunft
Operation teilfinanziert von der Europäischen Union
Europäischer Fonds für regionale Entwicklung
REPUBLIKA SLOVENIJA
MINISTRSTVO ZA GOSPODARSKI
RAZVOJ IN TEHNOLOGIJO

BASICS

Using rosls

- It allows you to directly in a package, stack, or common location by name rather than by package path.

- Usage:

```
$rosls [locationname[/subdir]]
```

- Try:

```
$rosls roscpp_tutorials
```

- It returns to you:

```
bin  cmake  manifest.xml  srv
```

- ROS allows you also TAB completion. For example:

```
$rosls roscpp_tut + TAB button
$rosls roscpp_tutorials
```

Naložba v vašo prihodnost
Operacijo delno financira Evropska unija
Evropski sklad za regionalni razvoj
Investition in Ihre Zukunft
Operation teilfinanziert von der Europäischen Union
Europäischer Fonds für regionale Entwicklung
REPUBLIKA SLOVENIJA
MINISTRSTVO ZA GOSPODARSKI RAZVOJ IN TEHNOLOGIJO

BASICS

Using roscreate

- All ROS packages consist of the many similar files : manifests, CMakeList.txt, mainpage.dox, and Makefiles. roscreate-pkg eliminates many tedious tasks of creating a new package by hand, and eliminates common errors caused by hand-typing build files and manifests.

- To create a new package in the current directory:

```
$roscreate-pkg [package_name]
```

- You can also specify dependencies of that package:

```
$roscreate-pkg [package_name] [depend1] [depend2]
```

Naložba v vašo prihodnost
Operacijo delno financira Evropska unija
Evropski sklad za regionalni razvoj
Investition in Ihre Zukunft
Operation teilfinanziert von der Europäischen Union
Europäischer Fonds für regionale Entwicklung
REPUBLIKA SLOVENIJA
MINISTRSTVO ZA GOSPODARSKI
RAZVOJ IN TEHNOLOGIJO

BASICS

Creating a ROS package

- Now go into your directory:

```
$cd ~/rosw/sandbox
```

```
$roscreate-pkg beginner_tutorials std_msgs rospy roscpp
```

- Now lets make sure that ROS can find your new package.

- Try moving to the directory for the package.

```
$roscd beginner_tutorials
```

```
$pwd
```

Naložba v vašo prihodnost
Operacijo delno financira Evropska unija
Evropski sklad za regionalni razvoj
Investition in Ihre Zukunft
Operation teilfinanziert von der Europäischen Union
Europäischer Fonds für regionale Entwicklung
REPUBLIKA SLOVENIJA
MINISTRSTVO ZA GOSPODARSKI
RAZVOJ IN TEHNOLOGIJO

BASICS

Package dependencies

- When using roscreate-pkg earlier, a few package dependencies were provided. These dependencies for a package are stored in the manifest file.

```
$rospack depends1 beginner_tutorials

std_msgs
rospy
roscpp
```

- Take a look at the manifest file:

```
$roscd beginner_tutorials
$cat manifest.xml
Or
$gedit manifest.xml
```

Naložba v vašo prihodnost
Operacijo delno financira Evropska unija
Evropski sklad za regionalni razvoj
Investition in Ihre Zukunft
Operation teilfinanziert von der Europäischen Union
Europäischer Fonds für regionale Entwicklung
REPUBLIKA SLOVENIJA
MINISTRSTVO ZA GOSPODARSKI RAZVOJ IN TEHNOLOGIJO

BASICS

Using rosmake

- When you type rosmake beginner_tutorials, it builds the beginner_tutorials package, plus every package that it depends on, in the correct order.

```
$rosmake [package]
```

- Try:

```
$rosmake beginner_tutorials
```

- We can also use rosmake to build multiple packages at once:

```
$rosmake [package1] [package2] [package3]
```

Naložba v vašo prihodnost
Operacijo delno financira Evropska unija
Evropski sklad za regionalni razvoj
Investition in Ihre Zukunft
Operation teilfinanziert von der Europäischen Union
Europäischer Fonds für regionale Entwicklung
REPUBLIKA SLOVENIJA
MINISTRSTVO ZA GOSPODARSKI RAZVOJ IN TEHNOLOGIJO

BASICS

Summary

- `$rosws:` setting up your workspace
- `$roscd:` navigation in ros packages
- `$rosls:` list of files in package/folder
- `$roscreate-pkg:` create new empty ROS package
- `$rospack:` handling packages
- `$rosmake:` build package

Naložba v vašo prihodnost
Operacijo delno financira Evropska unija
Evropski sklad za regionalni razvoj
Investition in Ihre Zukunft
Operation teilfinanziert von der Europäischen Union
Europäischer Fonds für regionale Entwicklung
REPUBLIKA SLOVENIJA
MINISTRSTVO ZA GOSPODARSKI
RAZVOJ IN TEHNOLOGIJO

BASICS

Exercise

1. Create ros package *test* in your working directory , with dependencies *roslib roscpp* and *beginner_tutorials*.
2. Build your package.
3. Find your newly created package using ros command.
4. List files that are in your package.
5. What dependency dose your package have, use ros command to list dependencies.

Naložba v vašo prihodnost
Operacijo delno financira Evropska unija
Evropski sklad za regionalni razvoj
Investition in Ihre Zukunft
Operation teilfinanziert von der Europäischen Union
Europäischer Fonds für regionale Entwicklung
REPUBLIKA SLOVENIJA
MINISTRSTVO ZA GOSPODARSKI
RAZVOJ IN TEHNOLOGIJO

ROS CONTENT

Naložba v vašo prihodnost
Operacijo delno financira Evropska unija
Evropski sklad za regionalni razvoj

Investition in Ihre Zukunft
Operation teilfinanziert von der Europäischen Union
Europäischer Fonds für regionale Entwicklung

REPUBLIKA SLOVENIJA
MINISTRSTVO ZA GOSPODARSKI
RAZVOJ IN TEHNOLOGIJO

NODES

Understanding ROS node

- **Nodes: A node is an executable that uses ROS to communicate with other nodes.**
- Messages: ROS data type used when subscribing or publishing to a topic.
- Topics: Nodes can *publish* messages to a topic as well as *subscribe* to a topic to receive messages.
- Master: Name service for ROS (i.e. helps nodes find each other)
- rosout: ROS equivalent of stdout/stderr
- roscore: Master + rosout + parameter server (parameter server will be introduced later)

Naložba v vašo prihodnost
Operacijo delno financira Evropska unija
Evropski sklad za regionalni razvoj
Investition in Ihre Zukunft
Operation teilfinanziert von der Europäischen Union
Europäischer Fonds für regionale Entwicklung
REPUBLIKA SLOVENIJA
MINISTRSTVO ZA GOSPODARSKI
RAZVOJ IN TEHNOLOGIJO

NODES

Understanding ROS node

- A node is an executable file within a ROS package.
- ROS nodes use a ROS client library to communicate with other nodes.
- Nodes can publish or subscribe to a Topic. Nodes can also provide or use a Service.

- ROS client libraries allow nodes written in different programming languages to communicate:
  - rospy = python client library
  - roscpp = c++ client library

Naložba v vašo prihodnost
Operacijo delno financira Evropska unija
Evropski sklad za regionalni razvoj
Investition in Ihre Zukunft
Operation teilfinanziert von der Europäischen Union
Europäischer Fonds für regionale Entwicklung
REPUBLIKA SLOVENIJA
MINISTRSTVO ZA GOSPODARSKI
RAZVOJ IN TEHNOLOGIJO

NODES

roscore

- For this tutorial we will use a lightweight simulator, please install it using:

```
$sudo apt-get install ros-groovy-ros-tutorials
```

- roscore is the first thing you should run when using ROS. Please run:

```
$roscore
```

- Open up a new terminal, and let's use rosnode. Rosnode displays information about the ROS nodes that are currently running. The rosnode list command lists these active nodes:

```
$rosnode list
```

- You will see:

  /rosout

- The rosnode info command returns information about a specific node.

```
$rosnode info /rosout
```

Naložba v vašo prihodnost
Operacijo delno financira Evropska unija
Evropski sklad za regionalni razvoj
Investition in Ihre Zukunft
Operation teilfinanziert von der Europäischen Union
Europäischer Fonds für regionale Entwicklung
REPUBLIKA SLOVENIJA
MINISTRSTVO ZA GOSPODARSKI
RAZVOJ IN TEHNOLOGIJO

NODES

rosrun

- Rosrun allows you to use the package name to directly run a node within a package (without having to know the package path).
- Usage:

```
$rosrun [package_name] [node_name]
```

- So now we can run the turtlesim_node in the turtlesim package. Then, in a **new terminal**:

```
$rosrun turtlesim turtlesim_node
```

- In a **new terminal**:

```
$rosnode list
```

- You will see something similar to:

```
/rosout
/turtlesim
```

Naložba v vašo prihodnost
Operacijo delno financira Evropska unija
Evropski sklad za regionalni razvoj
Investition in Ihre Zukunft
Operation teilfinanziert von der Europäischen Union
Europäischer Fonds für regionale Entwicklung
REPUBLIKA SLOVENIJA
MINISTRSTVO ZA GOSPODARSKI RAZVOJ IN TEHNOLOGIJO

NODES

rosrun

- One powerful feature of ROS is that you can reassign Names from the command-line. Close the turtlesim window to stop the node (or go back to the rosrun turtlesim terminal and use ctrl-C). Now let's re-run it, but this time use a Remapping Argument to change the node's name:

```
$rosrun turtlesim turtlesim_node __name:=my_turtle
```

- Now, if we go back and use rosnode list:

```
$rosnode list
```

- You will see something similar to:

```
/rosout
/my_turtle
```

- We see our new node. Let's use another rosnode command, to test that it's up:

```
$rosnode ping my_turtle
```

NODES

Summary

- `$rosnode list:` list of all active nodes
- `$rosnode info:` information of individual node
- `$rosrun:` run executable of specific package
- `$rosnode ping:` ping node to see if is responding/ is alive

Naložba v vašo prihodnost
Operacijo delno financira Evropska unija
Evropski sklad za regionalni razvoj
Investition in Ihre Zukunft
Operation teilfinanziert von der Europäischen Union
Europäischer Fonds für regionale Entwicklung
REPUBLIKA SLOVENIJA
MINISTRSTVO ZA GOSPODARSKI
RAZVOJ IN TEHNOLOGIJO

NODES

Exercise

1. Run two more *turtlesim_node*, with new turtle names in new terminals.
2. Ping one of your turtles .
3. List all running node in your ros system.
4. Try to run vizualization tool called rviz in package rviz.
5. Kill all nodes and roscore in terminals.

Naložba v vašo prihodnost
Operacijo delno financira Evropska unija
Evropski sklad za regionalni razvoj
Investition in Ihre Zukunft
Operation teilfinanziert von der Europäischen Union
Europäischer Fonds für regionale Entwicklung
REPUBLIKA SLOVENIJA
MINISTRSTVO ZA GOSPODARSKI
RAZVOJ IN TEHNOLOGIJO

ROS CONTENT

- PRESENTATION
- BASIC
- NODE
- TOPICS and MESSAGES
- ROS and C++ (Simple Publisher and Subscriber)
- SERVICES and PARAMETERS
- ROS and C++ (Simple Service and Client)
- TOOLS

Naložba v vašo prihodnost
Operacijo delno financira Evropska unija
Evropski sklad za regionalni razvoj
Investition in Ihre Zukunft
Operation teilfinanziert von der Europäischen Union
Europäischer Fonds für regionale Entwicklung
REPUBLIKA SLOVENIJA
MINISTRSTVO ZA GOSPODARSKI
RAZVOJ IN TEHNOLOGIJO

TOPIC

- Nodes: A node is an executable that uses ROS to communicate with other nodes.
- **Topics: Nodes can publish messages to a topic as well as *subscribe* to a topic to receive messages.**
- Messages: ROS data type used when subscribing or publishing to a topic.
- Master: Name service for ROS (i.e. helps nodes find each other)
- rosout: ROS equivalent of stdout/stderr
- roscore: Master + rosout + parameter server (parameter server will be introduced later)

Naložba v vašo prihodnost
Operacijo delno financira Evropska unija
Evropski sklad za regionalni razvoj
Investition in Ihre Zukunft
Operation teilfinanziert von der Europäischen Union
Europäischer Fonds für regionale Entwicklung
REPUBLIKA SLOVENIJA
MINISTRSTVO ZA GOSPODARSKI
RAZVOJ IN TEHNOLOGIJO

TOPIC

rostopic

- The rostopic tool allows you to get information about ROS topics.

```
$rostopic -h
   rostopic bw      //display bandwidth used by topic
   rostopic echo    //print messages to screen
   rostopic hz      //display publishing rate of topic
   rostopic list    //print information about active topics
   rostopic pub     //publish data to topic
   rostopic type    //print topic type
```

Naložba v vašo prihodnost
Operacijo delno financira Evropska unija
Evropski sklad za regionalni razvoj
Investition in Ihre Zukunft
Operation teilfinanziert von der Europäischen Union
Europäischer Fonds für regionale Entwicklung
REPUBLIKA SLOVENIJA
MINISTRSTVO ZA GOSPODARSKI
RAZVOJ IN TEHNOLOGIJO

TOPIC

rostopic

- Run ros core, if not already running.

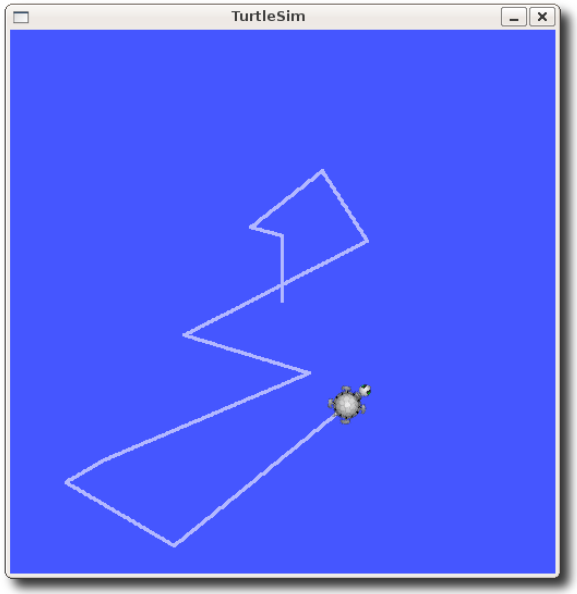
```
$roscore
```

- Run turtle sim node **in new terminal**.

```
$rosrun turtlesim turtlesim_node
```

- Run turtlesim teleop node **in new terminal**.

```
$rosrun turtlesim turtle_teleop_key
```




```
[ INFO] 1254264546.878445000: Started node [/teleop_turtle], pid [5528], bound
on [aqy], xmlrpc port [43918], tcpros port [55936], logging to [~/ros/ros/log
/teleop_turtle_5528.log], using [real] time
Reading from keyboard
---------------------------
Use arrow keys
```

Naložba v vašo prihodnost
Operacijo delno financira Evropska unija
Evropski sklad za regionalni razvoj
Investition in Ihre Zukunft
Operation teilfinanziert von der Europäischen Union
Europäischer Fonds für regionale Entwicklung
REPUBLIKA SLOVENIJA
MINISTRSTVO ZA GOSPODARSKI
RAZVOJ IN TEHNOLOGIJO

TOPIC

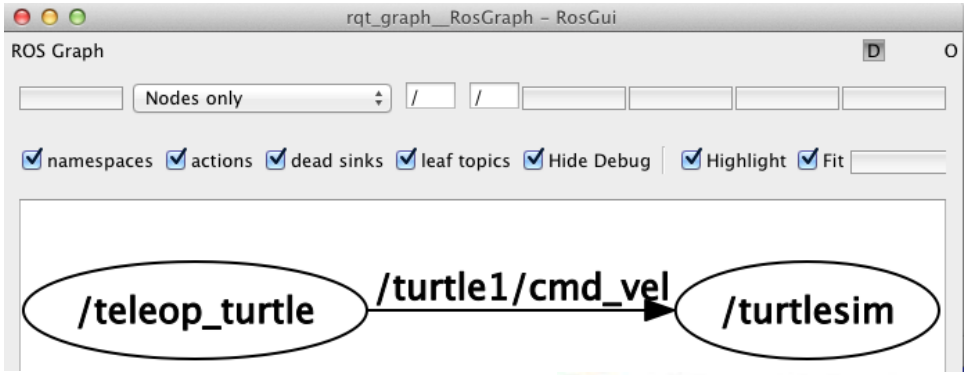
Using rqt_graph

- Tool rqt_graph creates a dynamic graph of what's going on in the system, rqt_graph is part of the rqt package. To install it, run:

```
$sudo apt-get install ros-groovy-rqt
$sudo apt-get install ros-groovy-rqt-common-plugins
```

- Run, in na new terminal:

```
$rosrun rqt_graph rqt_graph
```



Naložba v vašo prihodnost
Operacijo delno financira Evropska unija
Evropski sklad za regionalni razvoj
Investition in Ihre Zukunft
Operation teilfinanziert von der Europäischen Union
Europäischer Fonds für regionale Entwicklung
REPUBLIKA SLOVENIJA
MINISTRSTVO ZA GOSPODARSKI
RAZVOJ IN TEHNOLOGIJO

TOPIC

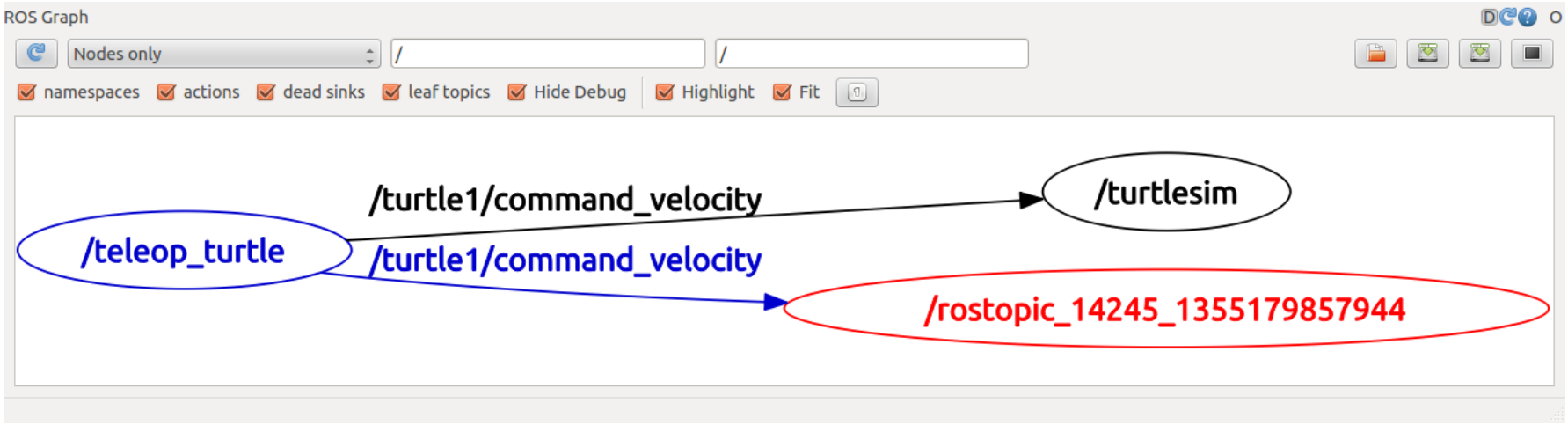
rostopic

- Command rostopic echo shows the data published on a topic. Usage:

```
$rostopic echo [topic]
```

- Let's look at the data published on the /turtle1/command_velocity topic by the turtle_teleop_key node, in a new terminal:

```
$rostopic echo /turtle1/command_velocity
```



Naložba v vašo prihodnost
Operacijo delno financira Evropska unija
Evropski sklad za regionalni razvoj
Investition in Ihre Zukunft
Operation teilfinanziert von der Europäischen Union
Europäischer Fonds für regionale Entwicklung
REPUBLIKA SLOVENIJA
MINISTRSTVO ZA GOSPODARSKI
RAZVOJ IN TEHNOLOGIJO

TOPIC

rostopic

- Now you should see the following when you press the up arrow key:

```
---
linear: 2.0
angular: 0.0
---
linear: 2.0
angular: 0.0
---
linear: 2.0
angular: 0.0
---
linear: 2.0
angular: 0.0
---
linear: 2.0
angular: 0.0
```

TOPIC

rostopic list

- Command rostopic list returns a list of all topics currently subscribed to and published.

- Lets figure out what argument the list sub-command needs. In a new terminal run:

```
$rostopic list -h
```

```
Usage: rostopic list [/topic]

Options:
  -h, --help            show this help message and exit
  -b BAGFILE, --bag=BAGFILE
                        list topics in .bag file
  -v, --verbose         list full details about each topic
  -p                    list only publishers
  -s                    list only subscribers
```

Naložba v vašo prihodnost
Operacijo delno financira Evropska unija
Evropski sklad za regionalni razvoj
Investition in Ihre Zukunft
Operation teilfinanziert von der Europäischen Union
Europäischer Fonds für regionale Entwicklung
REPUBLIKA SLOVENIJA
MINISTRSTVO ZA GOSPODARSKI RAZVOJ IN TEHNOLOGIJO

TOPIC

rostopic list

- For rostopic list use the verbose option. This displays a verbose list of topics to publish to and subscribe to and their type.

```
$rostopic list -v
```

```
Published topics:
 * /turtle1/color_sensor [turtlesim/Color] 1 publisher
 * /turtle1/command_velocity [turtlesim/Velocity] 1 publisher
 * /rosout [roslib/Log] 2 publishers
 * /rosout_agg [roslib/Log] 1 publisher
 * /turtle1/pose [turtlesim/Pose] 1 publisher

Subscribed topics:
 * /turtle1/command_velocity [turtlesim/Velocity] 1 subscriber
 * /rosout [roslib/Log] 1 subscriber
```

Naložba v vašo prihodnost
Operacijo delno financira Evropska unija
Evropski sklad za regionalni razvoj
Investition in Ihre Zukunft
Operation teilfinanziert von der Europäischen Union
Europäischer Fonds für regionale Entwicklung
REPUBLIKA SLOVENIJA
MINISTRSTVO ZA GOSPODARSKI RAZVOJ IN TEHNOLOGIJO

MESSAGES

- Nodes: A node is an executable that uses ROS to communicate with other nodes.
- Topics: Nodes can publish messages to a topic as well as *subscribe* to a topic to receive messages.
- **Messages: ROS data type used when subscribing or publishing to a topic.**
- Master: Name service for ROS (i.e. helps nodes find each other)
- rosout: ROS equivalent of stdout/stderr
- roscore: Master + rosout + parameter server (parameter server will be introduced later)

Naložba v vašo prihodnost
Operacijo delno financira Evropska unija
Evropski sklad za regionalni razvoj
Investition in Ihre Zukunft
Operation teilfinanziert von der Europäischen Union
Europäischer Fonds für regionale Entwicklung
REPUBLIKA SLOVENIJA
MINISTRSTVO ZA GOSPODARSKI
RAZVOJ IN TEHNOLOGIJO

MESSAGES

rostopic type

- Communication on topics happens by sending ROS messages between nodes. For the publisher (turtle_teleop_key) and subscriber (turtlesim_node) to communicate, the publisher and subscriber must send and receive the same type of message. This means that a topic type is defined by the message type published on it. The type of the message sent on a topic can be determined using rostopic type.

- Command rostopic type returns the message type of any topic being published.

```
$rostopic type [topic]
```

- Try:

```
$rostopic type /turtle1/command_velocity
```

- You should get:

```
turtlesim/Velocity
```

Naložba v vašo prihodnost
Operacijo delno financira Evropska unija
Evropski sklad za regionalni razvoj
Investition in Ihre Zukunft
Operation teilfinanziert von der Europäischen Union
Europäischer Fonds für regionale Entwicklung
REPUBLIKA SLOVENIJA
MINISTRSTVO ZA GOSPODARSKI
RAZVOJ IN TEHNOLOGIJO

MESSAGES

rosmsg show

- We can look at the details of the message using rosmsg:

```
$rosmsg show [topic]
```

- Run in a new terminal:

```
$rosmsg show turtlesim/Velocity
```

```
float32 linear
float32 angular
```

Naložba v vašo prihodnost
Operacijo delno financira Evropska unija
Evropski sklad za regionalni razvoj
Investition in Ihre Zukunft
Operation teilfinanziert von der Europäischen Union
Europäischer Fonds für regionale Entwicklung
REPUBLIKA SLOVENIJA
MINISTRSTVO ZA GOSPODARSKI
RAZVOJ IN TEHNOLOGIJO

MESSAGES

rostopic pub

- Command rostopic pub publishes data on to a topic currently advertised.

```
$rostopic pub [topic] [msg_type] [args]
```

- For example, this command will send a single message to turtlesim telling it to move with an linear velocity of 2.0, and an angular velocity of 1.8 :

```
$rostopic pub -1 /turtle1/command_velocity
   turtlesim/Velocity -- 2.0 1.8
```

Naložba v vašo prihodnost
Operacijo delno financira Evropska unija
Evropski sklad za regionalni razvoj
Investition in Ihre Zukunft
Operation teilfinanziert von der Europäischen Union
Europäischer Fonds für regionale Entwicklung
REPUBLIKA SLOVENIJA
MINISTRSTVO ZA GOSPODARSKI
RAZVOJ IN TEHNOLOGIJO

MESSAGES

rostopic pub

```
$rostopic pub -1 /turtle1/command_velocity turtlesim/Velocity -- 2.0 1.8
```

- is a pretty complicated example, so lets look at each argument in detail.

>> rostopic pub command will publish messages to a given topic

>> -1 (dash-one) option causes rostopic to only publish one message then exit

>> /turtle1/command_velocity is the name of the topic to publish to

>> turtlesim/Velocity is the message type to use when publishing the topic

>> -- double-dash tells the option parser that none of the following arguments is an option. This is required in cases where your arguments have a leading dash - (such as with negative numbers).

>> 2.0 1.8 is a turtlesim/Velocity msg has two floating point elements: linear and angular.

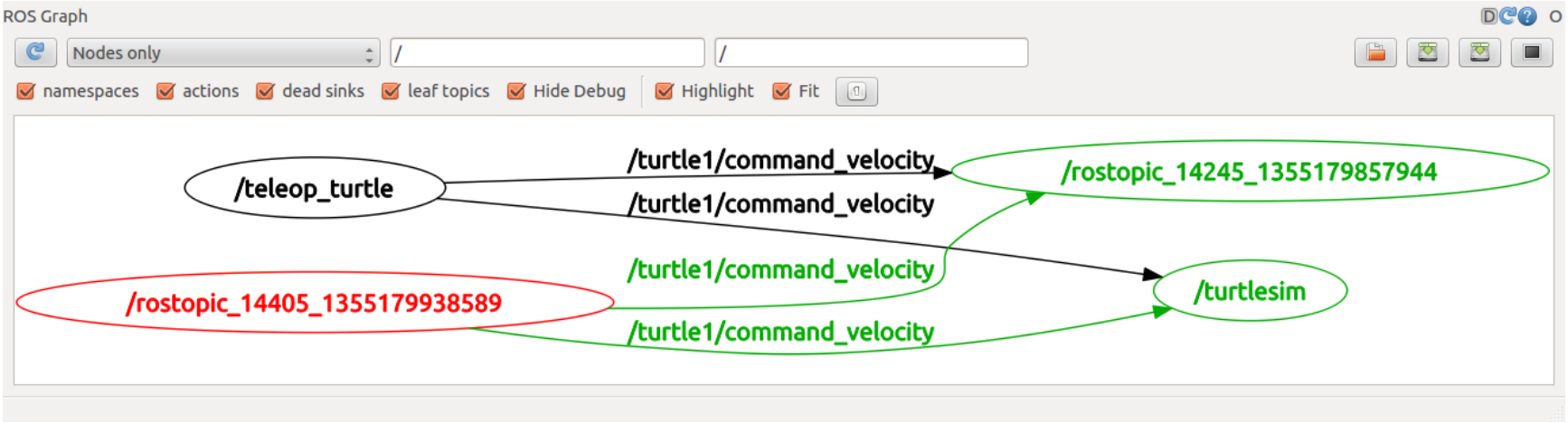
Naložba v vašo prihodnost
Operacijo delno financira Evropska unija
Evropski sklad za regionalni razvoj
Investition in Ihre Zukunft
Operation teilfinanziert von der Europäischen Union
Europäischer Fonds für regionale Entwicklung
REPUBLIKA SLOVENIJA
MINISTRSTVO ZA GOSPODARSKI
RAZVOJ IN TEHNOLOGIJO

MESSAGES

rostopic pub

- You may have noticed that the turtle has stopped moving; this is because the turtle requires a steady stream of commands at 1 Hz to keep moving. We can publish a steady stream of commands using rostopic pub -r command:

```
$rostopic pub /turtle1/command_velocity turtlesim/Velocity -r 1 -- 2.0 -1.8
```



Naložba v vašo prihodnost
Operacijo delno financira Evropska unija
Evropski sklad za regionalni razvoj
Investition in Ihre Zukunft
Operation teilfinanziert von der Europäischen Union
Europäischer Fonds für regionale Entwicklung
REPUBLIKA SLOVENIJA
MINISTRSTVO ZA GOSPODARSKI
RAZVOJ IN TEHNOLOGIJO

MESSAGES

rostopic hz

- Command rostopic hz reports the rate at which data is published

```
$rostopic hz [topic]
```

- Let's see how fast the turtlesim_node is publishing /turtle1/pose:

```
$rostopic hz /turtle1/pose
```

```
subscribed to [/turtle1/pose]
average rate: 59.354
        min: 0.005s max: 0.027s std dev: 0.00284s window: 58
average rate: 59.459
        min: 0.005s max: 0.027s std dev: 0.00271s window: 118
average rate: 59.539
        min: 0.004s max: 0.030s std dev: 0.00339s window: 177
average rate: 59.492
        min: 0.004s max: 0.030s std dev: 0.00380s window: 237
average rate: 59.463
        min: 0.004s max: 0.030s std dev: 0.00380s window: 290
```

- Now we can tell that the turtlesim is publishing data about our turtle at the rate of 60 Hz. We can also use rostopic type in conjunction with rosmsg show to get in depth information about a topic:

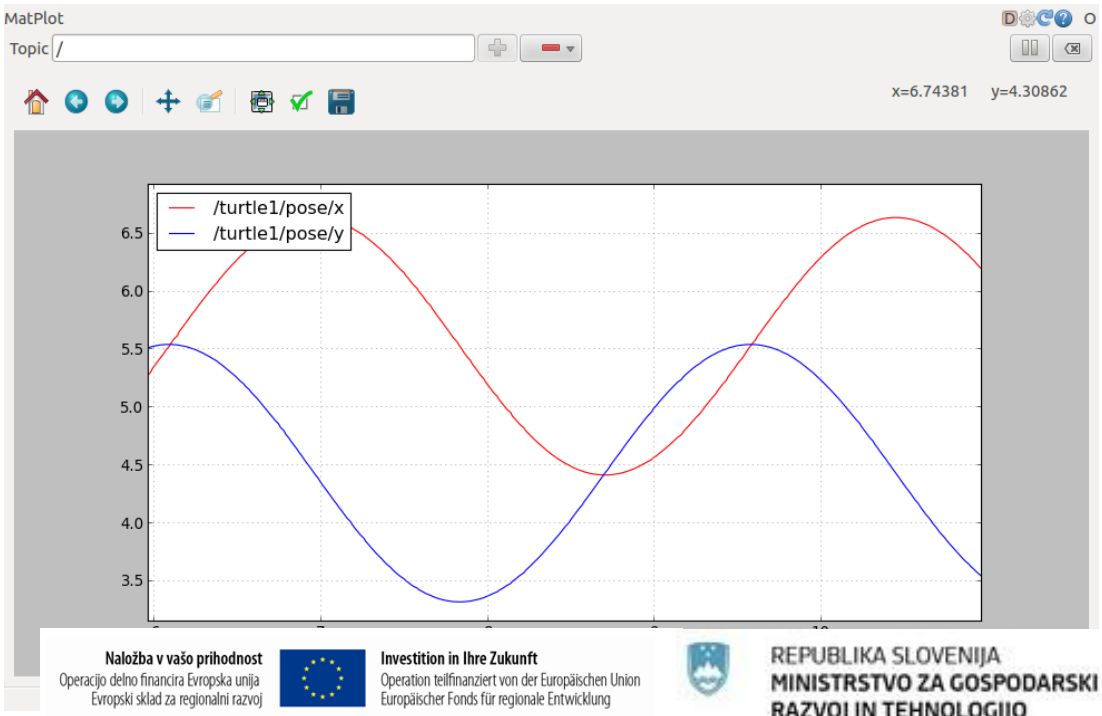
```
$rostopic rosmsg show
```

MESSAGES

Using rqt_plot

- Tool rqt_plot displays a scrolling time plot of the data published on topics. Here we'll use rqt_plot to plot the data being published on the /turtle1/pose topic.
- Start rqt_plot by typing in a new terminal:

```
$rosrun rqt_plot rqt_plot
```



MESSAGES

Summary

- `$rostopic echo:` show what data flow on specified topic
- `$rostopic list:` list of all active topics
- `$rostopic type:` data type of specific topic
- `$rosmsg show:` show rostopic message content
- `$rostopic pub:` publish commands on specified topic
- `$rostopic hz:` find out frequency of specific topic
- `$rqt_graph:` show node and topic graph structure
- `$rqt_plot:` plot specified topic

Naložba v vašo prihodnost
Operacijo delno financira Evropska unija
Evropski sklad za regionalni razvoj
Investition in Ihre Zukunft
Operation teilfinanziert von der Europäischen Union
Europäischer Fonds für regionale Entwicklung
REPUBLIKA SLOVENIJA
MINISTRSTVO ZA GOSPODARSKI
RAZVOJ IN TEHNOLOGIJO

MESSAGES

Exercise

1. Kill all active nodes.
2. Run roscore.
3. Run next command `$roslaunch stage hztest.xml`.
4. List all active nodes.
5. List all active topics.
6. Print data that is published in topic */base_scan*?
7. What is data type of topic */base_scan*?
8. At what frequency is published topic */odom*?
9. Close all terminals.

Naložba v vašo prihodnost
Operacijo delno financira Evropska unija
Evropski sklad za regionalni razvoj
Investition in Ihre Zukunft
Operation teilfinanziert von der Europäischen Union
Europäischer Fonds für regionale Entwicklung
REPUBLIKA SLOVENIJA
MINISTRSTVO ZA GOSPODARSKI
RAZVOJ IN TEHNOLOGIJO

ROS CONTENT

PRESENTATION

BASIC

NODE

TOPICS and MESSAGES

ROS and C++ (Simple Publisher and Subscriber)

SERVICES and PARAMETERS

ROS and C++ (Simple Service and Client)

TOOLS

Naložba v vašo prihodnost
Operacijo delno financira Evropska unija
Evropski sklad za regionalni razvoj

Investition in Ihre Zukunft
Operation teilfinanziert von der Europäischen Union
Europäischer Fonds für regionale Entwicklung

REPUBLIKA SLOVENIJA
MINISTRSTVO ZA GOSPODARSKI
RAZVOJ IN TEHNOLOGIJO

ROS and C++

rosed

- Command rosed is part of the rosbash suite. It allows you to directly edit a file within a package by using the package name rather than having to type the entire path to the package:
- Usage:

```
$rosed [package_name] [filename]
```

- Example (sudo apt-get install vim):

```
$rosed roscpp Logger.msg
```

- Using autocomplete:

```
$rosed [package_name] <tab>
```

- The default editor for rosed is vim, To set the default editor to something else edit your ~/.bashrc file to include or you can use gedit:

```
$gedit [filename]
```

Naložba v vašo prihodnost
Operacijo delno financira Evropska unija
Evropski sklad za regionalni razvoj
Investition in Ihre Zukunft
Operation teilfinanziert von der Europäischen Union
Europäischer Fonds für regionale Entwicklung
REPUBLIKA SLOVENIJA
MINISTRSTVO ZA GOSPODARSKI RAZVOJ IN TEHNOLOGIJO

ROS and C++

Simple Publisher and Subscriber

- This tutorial covers how to write a publisher and subscriber node in C++.
- "Node" is the ROS term for an executable that is connected to the ROS network. Here we'll create a publisher ("talker") node which will continually broadcast a message.

- Let‘s go to our beginer_tutorials directory:

```
$roscd beginner_tutorials
```

- In folder src/ create file talker.cpp, you can download talker.cpp file from: https://raw.github.com/ros/ros_tutorials/groovy-devel/roscpp_tutorials/talker/talker.cpp

- Lets take a look into code:

Naložba v vašo prihodnost
Operacijo delno financira Evropska unija
Evropski sklad za regionalni razvoj
Investition in Ihre Zukunft
Operation teilfinanziert von der Europäischen Union
Europäischer Fonds für regionale Entwicklung
REPUBLIKA SLOVENIJA
MINISTRSTVO ZA GOSPODARSKI
RAZVOJ IN TEHNOLOGIJO

ROS and C++

Simple Publisher

#include "ros/ros.h“

- Include that includes all the headers necessary to use the most common public pieces of the ROS system.

#include "std_msgs/String.h“

- This includes the std_msgs/String message, which resides in the std_msgs package. This is a header generated automatically from the String.msg file in that package.

ros::init(argc, argv, "talker");

- Initialize ROS.

ros::NodeHandle n;

- Create a handle to this process' node. The first NodeHandle created will actually do the initialization of the node, and the last one destructed will cleanup any resources the node was using.

Naložba v vašo prihodnost
Operacijo delno financira Evropska unija
Evropski sklad za regionalni razvoj
Investition in Ihre Zukunft
Operation teilfinanziert von der Europäischen Union
Europäischer Fonds für regionale Entwicklung
REPUBLIKA SLOVENIJA
MINISTRSTVO ZA GOSPODARSKI RAZVOJ IN TEHNOLOGIJO

ROS and C++

Simple Publisher

ros::Publisher chatter_pub = n.advertise<std_msgs::String>("chatter", 1000);

- Tell the master that we are going to be publishing a message of type std_msgs/String on the topic chatter. This lets the master tell any nodes listening on chatter that we are going to publish data on that topic. The second argument is the size of our publishing queue. In this case if we are publishing too quickly it will buffer up a maximum of 1000 messages before beginning to throw away old ones.
- NodeHandle::advertise() returns a ros::Publisher object, which serves two purposes: 1) it contains a publish() method that lets you publish messages onto the topic it was created with, and 2) when it goes out of scope, it will automatically unadvertise.

ros::Rate loop_rate(10);

- allows you to specify a frequency that you would like to loop at.

Naložba v vašo prihodnost
Operacijo delno financira Evropska unija
Evropski sklad za regionalni razvoj
Investition in Ihre Zukunft
Operation teilfinanziert von der Europäischen Union
Europäischer Fonds für regionale Entwicklung
REPUBLIKA SLOVENIJA
MINISTRSTVO ZA GOSPODARSKI
RAZVOJ IN TEHNOLOGIJO

ROS and C++

Simple Publisher

```
int count = 0;
while (ros::ok())
{
```

- Loop untill Ctrl+C handling.

```
std_msgs::String msg;
std::stringstream ss;
ss << "hello world " << count;
msg.data = ss.str();
```

- We broadcast a message on ROS using a message-adapted class, generally generated from a msg file. More complicated datatypes are possible, but for now we're going to use the standard String message, which has one member: "data"

Naložba v vašo prihodnost
Operacijo delno financira Evropska unija
Evropski sklad za regionalni razvoj
Investition in Ihre Zukunft
Operation teilfinanziert von der Europäischen Union
Europäischer Fonds für regionale Entwicklung
REPUBLIKA SLOVENIJA
MINISTRSTVO ZA GOSPODARSKI RAZVOJ IN TEHNOLOGIJO

ROS and C++

Simple Publisher

chatter_pub.publish(msg);

- Now we actually broadcast the message to anyone who is connected.

ROS_INFO("%s", msg.data.c_str());

- ROS_INFO and friends are our replacement for printf/cout.

ros::spinOnce();

- For trigering callbacks, not needed in this program.

loop_rate.sleep();

- Now we use the ros::Rate object to sleep for the time remaining to let us hit our 10hz publish rate.

Naložba v vašo prihodnost
Operacijo delno financira Evropska unija
Evropski sklad za regionalni razvoj
Investition in Ihre Zukunft
Operation teilfinanziert von der Europäischen Union
Europäischer Fonds für regionale Entwicklung
REPUBLIKA SLOVENIJA
MINISTRSTVO ZA GOSPODARSKI
RAZVOJ IN TEHNOLOGIJO

ROS and C++

Simple Publisher - SUMMARY

What have we done:

- Initialize the ROS system.
- Advertise that we are going to be publishing std_msgs/String messages on the chatter topic to the master.
- Loop while publishing messages to chatter 10 times a second.

Now we need to write a node to receive the messages.

Naložba v vašo prihodnost
Operacijo delno financira Evropska unija
Evropski sklad za regionalni razvoj
Investition in Ihre Zukunft
Operation teilfinanziert von der Europäischen Union
Europäischer Fonds für regionale Entwicklung
REPUBLIKA SLOVENIJA
MINISTRSTVO ZA GOSPODARSKI
RAZVOJ IN TEHNOLOGIJO

ROS and C++

Simple Subscriber

- In folder beginner_tutorials/src/ create file talker.cpp, you can download talker.cpp file from (use wget):

https://raw.github.com/ros/ros_tutorials/groovy-devel/roscpp_tutorials/listener/listener.cpp

- Lets take a look into code:

```
void chatterCallback(const std_msgs::String::ConstPtr& msg)
{
      ROS_INFO("I heard: [%s]", msg->data.c_str());
}
```

- This is the callback function that will get called when a new message has arrived on the chatter topic.

Naložba v vašo prihodnost
Operacijo delno financira Evropska unija
Evropski sklad za regionalni razvoj
Investition in Ihre Zukunft
Operation teilfinanziert von der Europäischen Union
Europäischer Fonds für regionale Entwicklung
REPUBLIKA SLOVENIJA
MINISTRSTVO ZA GOSPODARSKI
RAZVOJ IN TEHNOLOGIJO

ROS and C++

Simple Subscriber

ros::Subscriber sub = n.subscribe("chatter", 1000, chatterCallback);

- Subscribe to the chatter topic with the master. ROS will call the chatterCallback() function whenever a new message arrives. The 2nd argument is the queue size, in case we are not able to process messages fast enough. In this case, if the queue reaches 1000 messages, we will start throwing away old messages as new ones arrive.
- NodeHandle::subscribe() returns a ros::Subscriber object, that you must hold on to until you want to unsubscribe. When the Subscriber object is destructed, it will automatically unsubscribe from the chatter topic.

ros::spin();

- ros::spin() enters a loop, calling message callbacks as fast as possible. Don't worry though, if there's nothing for it to do it won't use much CPU. ros::spin() will exit once ros::ok() returns false, which means ros::shutdown() has been called, either by the default Ctrl-C handler, the master telling us to shutdown, or it being called manually

Naložba v vašo prihodnost
Operacijo delno financira Evropska unija
Evropski sklad za regionalni razvoj
Investition in Ihre Zukunft
Operation teilfinanziert von der Europäischen Union
Europäischer Fonds für regionale Entwicklung
REPUBLIKA SLOVENIJA
MINISTRSTVO ZA GOSPODARSKI
RAZVOJ IN TEHNOLOGIJO

ROS and C++

Simple Subscriber - SUMMARY

We have now:

- Initialize the ROS system
- Subscribe to the chatter topic
- Spin, waiting for messages to arrive
- When a message arrives, the chatterCallback() function is called

Lets build our nodes!

Naložba v vašo prihodnost
Operacijo delno financira Evropska unija
Evropski sklad za regionalni razvoj
Investition in Ihre Zukunft
Operation teilfinanziert von der Europäischen Union
Europäischer Fonds für regionale Entwicklung
REPUBLIKA SLOVENIJA
MINISTRSTVO ZA GOSPODARSKI
RAZVOJ IN TEHNOLOGIJO

ROS and C++

Simple Publisher and Subscriber – building nodes

- Go to your beginnger tutorials folder:

```
$roscd beginer_tutorial
```

- Open whit rosed or gedit CMakeList.txt and add following lines:

```
$gedit CMakeList.txt
```

```
rosbuild_add_executable(talker src/talker.cpp)
rosbuild_add_executable(listener src/listener.cpp)
```

- Build our package:

```
$rosmake beginner_tutorials
```

Naložba v vašo prihodnost
Operacijo delno financira Evropska unija
Evropski sklad za regionalni razvoj
Investition in Ihre Zukunft
Operation teilfinanziert von der Europäischen Union
Europäischer Fonds für regionale Entwicklung
REPUBLIKA SLOVENIJA
MINISTRSTVO ZA GOSPODARSKI
RAZVOJ IN TEHNOLOGIJO

ROS and C++

Simple Publisher and Subscriber – building nodes

- Make sure we are running roscore:

```
$rocore
```

- Run talker node:

```
$rosrun beginner_tutorials talker
```

- Run listener node:

```
$rosrun beginner_tutorials listener
```

Naložba v vašo prihodnost
Operacijo delno financira Evropska unija
Evropski sklad za regionalni razvoj
Investition in Ihre Zukunft
Operation teilfinanziert von der Europäischen Union
Europäischer Fonds für regionale Entwicklung
REPUBLIKA SLOVENIJA
MINISTRSTVO ZA GOSPODARSKI
RAZVOJ IN TEHNOLOGIJO

ROS and C++

Summary

`$rosed:`	opens default editor for editing files
`$gedit:`	simple editor
`$talker:`	publish data on topic /chatter
`$listener:`	listen topic /chatter

Naložba v vašo prihodnost
Operacijo delno financira Evropska unija
Evropski sklad za regionalni razvoj
Investition in Ihre Zukunft
Operation teilfinanziert von der Europäischen Union
Europäischer Fonds für regionale Entwicklung
REPUBLIKA SLOVENIJA
MINISTRSTVO ZA GOSPODARSKI
RAZVOJ IN TEHNOLOGIJO

ROS and C++

Exercise

1. Close all active nodes, you can leave roscore running.
2. Modify program so that the message topic will be /speaker. Check rgt_graph structure.
3. Modify program so that the message that is received by listener will be „hello [your name]“.

Naložba v vašo prihodnost
Operacijo delno financira Evropska unija
Evropski sklad za regionalni razvoj
Investition in Ihre Zukunft
Operation teilfinanziert von der Europäischen Union
Europäischer Fonds für regionale Entwicklung
REPUBLIKA SLOVENIJA
MINISTRSTVO ZA GOSPODARSKI
RAZVOJ IN TEHNOLOGIJO

ROS CONTENT

- PRESENTATION
- BASIC
- NODE
- TOPICS and MESSAGES
- ROS and C++ (Simple Publisher and Subscriber)
- SERVICES and PARAMETERS
- ROS and C++ (Simple Service and Client)
- TOOLS

Naložba v vašo prihodnost
Operacijo delno financira Evropska unija
Evropski sklad za regionalni razvoj
Investition in Ihre Zukunft
Operation teilfinanziert von der Europäischen Union
Europäischer Fonds für regionale Entwicklung
REPUBLIKA SLOVENIJA
MINISTRSTVO ZA GOSPODARSKI
RAZVOJ IN TEHNOLOGIJO

SERVICES and PARAMETERS

rosservice and rosparam

This tutorial introduces ROS services, and parameters as well as using the rosservice and rosparam command line tools.

Services are another way that nodes can communicate with each other. Services allow nodes to send a **request** and receive a **response**.

- Let‘s run turtle_sim node:

```
$rosrun turtlesim turtlesim_node
```

- Usage: `$rosservice`

```
$rosservice list    //print information about active services
$rosservice call    //call the service with the provided args
$rosservice type    //print service type
$rosservice find    //find services by service type
$rosservice uri     //print service ROSRPC uri
```

Naložba v vašo prihodnost
Operacijo delno financira Evropska unija
Evropski sklad za regionalni razvoj
Investition in Ihre Zukunft
Operation teilfinanziert von der Europäischen Union
Europäischer Fonds für regionale Entwicklung
REPUBLIKA SLOVENIJA
MINISTRSTVO ZA GOSPODARSKI RAZVOJ IN TEHNOLOGIJO

SERVICES and PARAMETERS

rosservice and rosparam

- let's look at what services the turtlesim provides:

```
$rosservice list
```

```
/clear
/kill
/reset
/rosout/get_loggers
/rosout/set_logger_level
/spawn
/teleop_turtle/get_loggers
/teleop_turtle/set_logger_level
/turtle1/set_pen
/turtle1/teleport_absolute
/turtle1/teleport_relative
/turtlesim/get_loggers
/turtlesim/set_logger_level
```

Naložba v vašo prihodnost
Operacijo delno financira Evropska unija
Evropski sklad za regionalni razvoj
Investition in Ihre Zukunft
Operation teilfinanziert von der Europäischen Union
Europäischer Fonds für regionale Entwicklung
REPUBLIKA SLOVENIJA
MINISTRSTVO ZA GOSPODARSKI
RAZVOJ IN TEHNOLOGIJO

SERVICES and PARAMETERS

rosservice and rosparam

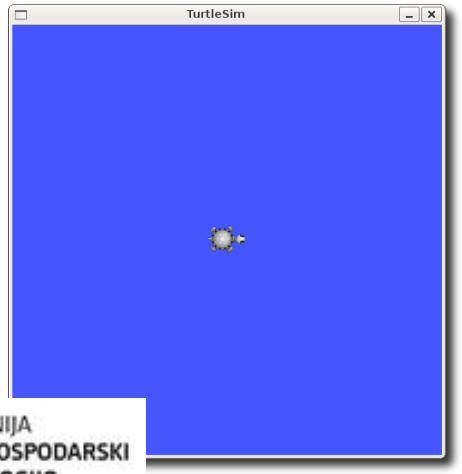
- Let's find out what type the clear service is:

```
$rosservice type /clear
```

```
std_srvs/Empty
```

- This service is empty, this means when the service call is made it takes no arguments (i.e. it sends no data when making a **request** and receives no data when receiving a **response**). Let's call this service using rosservice call:

```
$rosservice call clear
```



- This service clears background of turtlesim.

Naložba v vašo prihodnost
Operacijo delno financira Evropska unija
Evropski sklad za regionalni razvoj
Investition in Ihre Zukunft
Operation teilfinanziert von der Europäischen Union
Europäischer Fonds für regionale Entwicklung
REPUBLIKA SLOVENIJA
MINISTRSTVO ZA GOSPODARSKI RAZVOJ IN TEHNOLOGIJO

SERVICES and PARAMETERS

rosservice and rosparam

- Let's look at the case where the service has arguments by looking at the information for the service spawn:

```
$rosservice type spawn| rossrv show
```

```
float32 x
float32 y
float32 theta
string name
---
string name
```

- This service lets us spawn a new turtle at a given location and orientation. The name field is optional, so let's not give our new turtle a name and let turtlesim create one for us.

```
$rosservice call spawn 2 2 0.2 ""
```

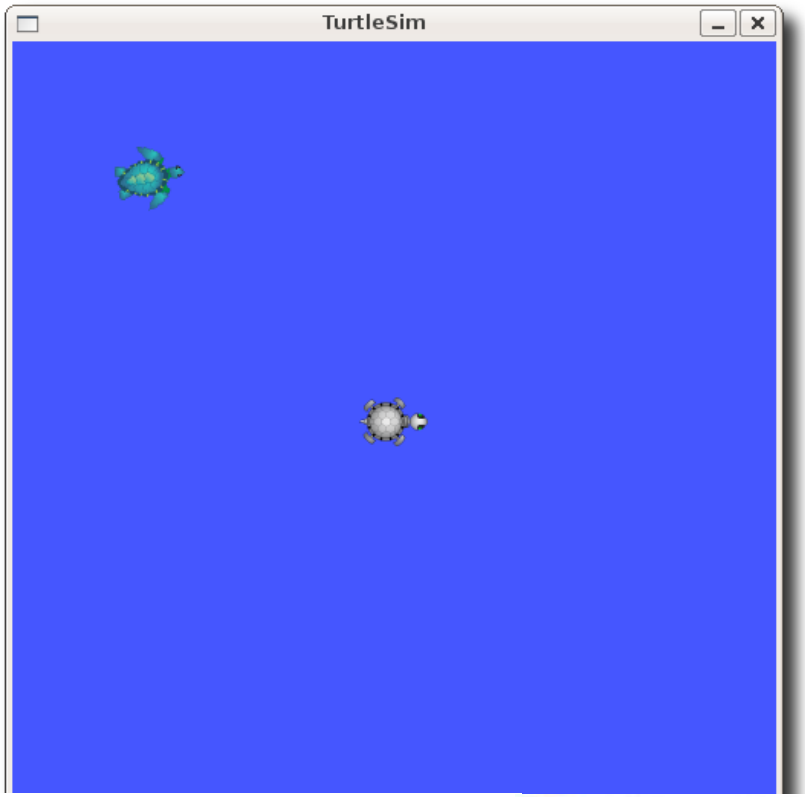
Naložba v vašo prihodnost
Operacijo delno financira Evropska unija
Evropski sklad za regionalni razvoj
Investition in Ihre Zukunft
Operation teilfinanziert von der Europäischen Union
Europäischer Fonds für regionale Entwicklung
REPUBLIKA SLOVENIJA
MINISTRSTVO ZA GOSPODARSKI
RAZVOJ IN TEHNOLOGIJO

SERVICES and PARAMETERS

rosservice and rosparam

- The service call returns with the name of the newly created turtle :

```
name: turtle2
```



Naložba v vašo prihodnost
Operacijo delno financira Evropska unija
Evropski sklad za regionalni razvoj
Investition in Ihre Zukunft
Operation teilfinanziert von der Europäischen Union
Europäischer Fonds für regionale Entwicklung
REPUBLIKA SLOVENIJA
MINISTRSTVO ZA GOSPODARSKI
RAZVOJ IN TEHNOLOGIJO

SERVICES and PARAMETERS

rosservice and rosparam

- Command rosparam allows you to store and manipulate data on the ROS Parameter Server.
- The Parameter Server can store integers, floats, boolean, dictionaries, and lists. rosparam has many commands that can be used on parameters, as shown below:

- Usage: `$rosparam`

```
$rosparam set       //set parameter
$rosparam get       //Get parameter
$rosparam load      //load parameters from file
$rosparam dump      //dump parameters to file
$rosparam delete    //delete parameter
$rosparam list      //list parameter names
```

Naložba v vašo prihodnost
Operacijo delno financira Evropska unija
Evropski sklad za regionalni razvoj
Investition in Ihre Zukunft
Operation teilfinanziert von der Europäischen Union
Europäischer Fonds für regionale Entwicklung
REPUBLIKA SLOVENIJA
MINISTRSTVO ZA GOSPODARSKI
RAZVOJ IN TEHNOLOGIJO

SERVICES and PARAMETERS

rosservice and rosparam

- List paramaters:

```
$rosparam list
```

```
/background_b
/background_g
/background_r
/roslaunch/uris/aqy:51932
/run_id
```

- Usage:

```
$rosparam set [param_name]
$rosparam get [param_name]
```

- Let's change one of the parameter values using rosparam set:

```
$rosparam set background_r 150
```

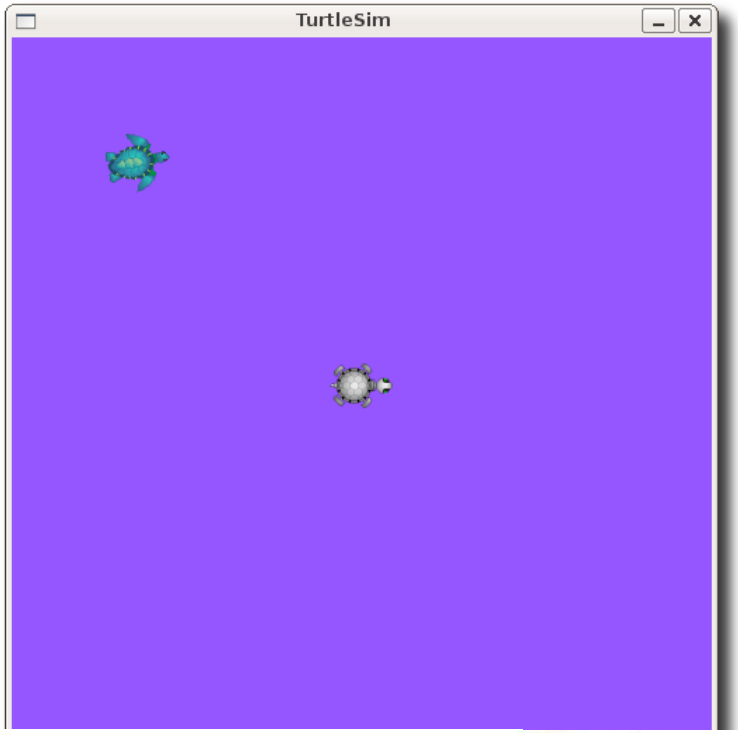
Naložba v vašo prihodnost
Operacijo delno financira Evropska unija
Evropski sklad za regionalni razvoj
Investition in Ihre Zukunft
Operation teilfinanziert von der Europäischen Union
Europäischer Fonds für regionale Entwicklung
REPUBLIKA SLOVENIJA
MINISTRSTVO ZA GOSPODARSKI RAZVOJ IN TEHNOLOGIJO

SERVICES and PARAMETERS

rosservice and rosparam

- This changes the parameter value, now we have to call the clear service for the parameter change to take effect:

```
$rosservice call clear
```



Naložba v vašo prihodnost
Operacijo delno financira Evropska unija
Evropski sklad za regionalni razvoj
Investition in Ihre Zukunft
Operation teilfinanziert von der Europäischen Union
Europäischer Fonds für regionale Entwicklung
REPUBLIKA SLOVENIJA
MINISTRSTVO ZA GOSPODARSKI
RAZVOJ IN TEHNOLOGIJO

SERVICES and PARAMETERS

rosservice and rosparam

- We can also use rosparam get / to show us the contents of the entire Parameter Server:

```
$rosparam get /
```

```
background_b: 255
background_g: 86
background_r: 150
roslaunch:
  uris: {'aqy:51932': 'http://aqy:51932/'}
run_id: e07ea71e-98df-11de-8875-001b21201aa8
```

- Or only one parameter:

```
$rosparam get background_g
```

```
86
```

Naložba v vašo prihodnost
Operacijo delno financira Evropska unija
Evropski sklad za regionalni razvoj
Investition in Ihre Zukunft
Operation teilfinanziert von der Europäischen Union
Europäischer Fonds für regionale Entwicklung
REPUBLIKA SLOVENIJA
MINISTRSTVO ZA GOSPODARSKI
RAZVOJ IN TEHNOLOGIJO

SERVICES and PARAMETERS

rosservice and rosparam

- You may wish to store this in a file so that you can reload it at another time. This is easy using rosparam:

- Usage:

```
$rosparam dump [file_name]
$rosparam load [file_name] [namespace]
```

- Here we write all parameters to the file params.yaml:

```
$rosparam dump params.yaml
```

Naložba v vašo prihodnost
Operacijo delno financira Evropska unija
Evropski sklad za regionalni razvoj
Investition in Ihre Zukunft
Operation teilfinanziert von der Europäischen Union
Europäischer Fonds für regionale Entwicklung
REPUBLIKA SLOVENIJA
MINISTRSTVO ZA GOSPODARSKI
RAZVOJ IN TEHNOLOGIJO

SERVICES and PARAMETERS

Summary

`$rosservice list:`	list all active services
`$rosservice type:`	show data type of specific srvice
`$rosservice call:`	call specific service whit parameters
`$rosparam list:`	list of avalible parameters
`$rosparam set:`	set specific parameter
`$rosparam get:`	get value of specific parameter
`$rosparam dump:`	save parameters to file
`$rosparam load:`	load parameters from file

Naložba v vašo prihodnost
Operacijo delno financira Evropska unija
Evropski sklad za regionalni razvoj
Investition in Ihre Zukunft
Operation teilfinanziert von der Europäischen Union
Europäischer Fonds für regionale Entwicklung
REPUBLIKA SLOVENIJA
MINISTRSTVO ZA GOSPODARSKI
RAZVOJ IN TEHNOLOGIJO

SERVICES and PARAMETERS

Exercise

1. Clear turtle path history.
2. Set turtle background color to R:255 B:125 G:50.

Naložba v vašo prihodnost
Operacijo delno financira Evropska unija
Evropski sklad za regionalni razvoj
Investition in Ihre Zukunft
Operation teilfinanziert von der Europäischen Union
Europäischer Fonds für regionale Entwicklung
REPUBLIKA SLOVENIJA
MINISTRSTVO ZA GOSPODARSKI
RAZVOJ IN TEHNOLOGIJO

ROS CONTENT

PRESENTATION

BASIC

NODE

TOPICS and MESSAGES

ROS and C++ (Simple Publisher and Subscriber)

SERVICES and PARAMETERS

ROS and C++ (Simple Service and Client)

TOOLS

Naložba v vašo prihodnost
Operacijo delno financira Evropska unija
Evropski sklad za regionalni razvoj

Investition in Ihre Zukunft
Operation teilfinanziert von der Europäischen Union
Europäischer Fonds für regionale Entwicklung

REPUBLIKA SLOVENIJA
MINISTRSTVO ZA GOSPODARSKI
RAZVOJ IN TEHNOLOGIJO

ROS and C++

Simple Service and Client

- First we will create service message.
- Second we will create service server.
- And final we will create client
- Run our new service.

Naložba v vašo prihodnost
Operacijo delno financira Evropska unija
Evropski sklad za regionalni razvoj
Investition in Ihre Zukunft
Operation teilfinanziert von der Europäischen Union
Europäischer Fonds für regionale Entwicklung
REPUBLIKA SLOVENIJA
MINISTRSTVO ZA GOSPODARSKI
RAZVOJ IN TEHNOLOGIJO

ROS and C++

Creating Service Message

- Let's use the package we just created to create a srv:

```
$roscd beginner_tutorials
$mkdir srv
```

- Instead of creating a new srv definition by hand, we will copy an existing one from another package.
- For that, roscp is a useful commandline tool for copying files from one package to another.
- Usage:

```
$roscp [package_name] [file_to_copy_path] [copy_path]
```

- Now we can copy a service from the rospy_tutorials package:

```
$roscp rospy_tutorials AddTwoInts.srv srv/AddTwoInts.srv
```

ROS and C++

Creating Service Message

- There's one more step, though. We need to make sure that the srv files are turned into source code for C++, Python, and other languages.
- Once again, open CMakeLists.txt and remove # to uncomment the following line:

```
# rosbuild_gensrv()
```

Naložba v vašo prihodnost
Operacijo delno financira Evropska unija
Evropski sklad za regionalni razvoj
Investition in Ihre Zukunft
Operation teilfinanziert von der Europäischen Union
Europäischer Fonds für regionale Entwicklung
REPUBLIKA SLOVENIJA
MINISTRSTVO ZA GOSPODARSKI
RAZVOJ IN TEHNOLOGIJO

ROS and C++

rossrv

- That's all you need to do to create a srv. Let's make sure that ROS can see it using the rossrv show command.
- Usage:

```
$rossrv show <service type>
```

- Example:

```
$rossrv show beginner_tutorials/AddTwoInts
```

- You will see:

```
int64 a
int64 b
---
int64 sum
```

Naložba v vašo prihodnost
Operacijo delno financira Evropska unija
Evropski sklad za regionalni razvoj
Investition in Ihre Zukunft
Operation teilfinanziert von der Europäischen Union
Europäischer Fonds für regionale Entwicklung
REPUBLIKA SLOVENIJA
MINISTRSTVO ZA GOSPODARSKI
RAZVOJ IN TEHNOLOGIJO

ROS and C++

rossrv

- Now that we have made some new messages we need to make our package againUsage:

```
$rosmake beginner_tutorials
```

Naložba v vašo prihodnost
Operacijo delno financira Evropska unija
Evropski sklad za regionalni razvoj
Investition in Ihre Zukunft
Operation teilfinanziert von der Europäischen Union
Europäischer Fonds für regionale Entwicklung
REPUBLIKA SLOVENIJA
MINISTRSTVO ZA GOSPODARSKI
RAZVOJ IN TEHNOLOGIJO

ROS and C++

Simple Service

- Here we'll create the service ("add_two_ints_server") node which will receive two ints and return the sum.
- Go to your beginner_tutorials:

```
$roscd beginner_tutorials
```

- Create the src/add_two_ints_server.cpp file within the beginner_tutorials package and paste code inside:

- Code can be found : http://wiki.ros.org/ROS/Tutorials/WritingServiceClient

```
$cd src
$touch add_two_ints_server.cpp
$gedit add_two_ints_server.cpp
```

Naložba v vašo prihodnost
Operacijo delno financira Evropska unija
Evropski sklad za regionalni razvoj
Investition in Ihre Zukunft
Operation teilfinanziert von der Europäischen Union
Europäischer Fonds für regionale Entwicklung
REPUBLIKA SLOVENIJA
MINISTRSTVO ZA GOSPODARSKI RAZVOJ IN TEHNOLOGIJO

ROS and C++

Simple Service

```
#include "ros/ros.h"
#include "beginner_tutorials/AddTwoInts.h"

bool add(beginner_tutorials::AddTwoInts::Request  &req,  beginner_tutorials::AddTwoInts::Response &res)
{
    res.sum = req.a + req.b;
    ROS_INFO("request: x=%ld, y=%ld", (long int)req.a, (long int)req.b);
    ROS_INFO("sending back response: [%ld]", (long int)res.sum);
    return true;
}
int main(int argc, char **argv)
{
    ros::init(argc, argv, "add_two_ints_server");
    ros::NodeHandle n;
    ros::ServiceServer service = n.advertiseService("add_two_ints", add);
    ROS_INFO("Ready to add two ints.");
    ros::spin();
    return 0;
}
```

Naložba v vašo prihodnost
Operacijo delno financira Evropska unija
Evropski sklad za regionalni razvoj
Investition in Ihre Zukunft
Operation teilfinanziert von der Europäischen Union
Europäischer Fonds für regionale Entwicklung
REPUBLIKA SLOVENIJA
MINISTRSTVO ZA GOSPODARSKI
RAZVOJ IN TEHNOLOGIJO

ROS and C++

Simple Service

- Now, let's break the code down:

```
#include "ros/ros.h"
#include "beginner_tutorials/AddTwoInts.h„
```

- beginner_tutorials/AddTwoInts.h is the header file generated from the srv file that we created earlier.

```
bool add(beginner_tutorials::AddTwoInts::Request  &req,
         beginner_tutorials::AddTwoInts::Response &res)
```

- This function provides the service for adding two ints, it takes in the request and response type defined in the srv file and returns a boolean.

Naložba v vašo prihodnost
Operacijo delno financira Evropska unija
Evropski sklad za regionalni razvoj
Investition in Ihre Zukunft
Operation teilfinanziert von der Europäischen Union
Europäischer Fonds für regionale Entwicklung
REPUBLIKA SLOVENIJA
MINISTRSTVO ZA GOSPODARSKI
RAZVOJ IN TEHNOLOGIJO

ROS and C++

Simple Service

```
{
  res.sum = req.a + req.b;
  ROS_INFO("request: x=%ld, y=%ld", (long int)req.a, (long int)req.b);
  ROS_INFO("sending back response: [%ld]", (long int)res.sum);
  return true;
}
```

- Here the two ints are added and stored in the response. Then some information about the request and response are logged. Finally the service returns true when it is complete.

```
ros::ServiceServer service = n.advertiseService("add_two_ints", add);
```

- Here the service is created and advertised over ROS.

Naložba v vašo prihodnost
Operacijo delno financira Evropska unija
Evropski sklad za regionalni razvoj
Investition in Ihre Zukunft
Operation teilfinanziert von der Europäischen Union
Europäischer Fonds für regionale Entwicklung
REPUBLIKA SLOVENIJA
MINISTRSTVO ZA GOSPODARSKI RAZVOJ IN TEHNOLOGIJO

ROS and C++

Simple Client

- Create the src/add_two_ints_client.cpp file within the beginner_tutorials package and paste the following inside it:

- Code can be found : http://wiki.ros.org/ROS/Tutorials/WritingServiceClient

```
$cd src
$touch add_two_ints_client.cpp
$gedit add_two_ints_client.cpp
```

Naložba v vašo prihodnost
Operacijo delno financira Evropska unija
Evropski sklad za regionalni razvoj
Investition in Ihre Zukunft
Operation teilfinanziert von der Europäischen Union
Europäischer Fonds für regionale Entwicklung
REPUBLIKA SLOVENIJA
MINISTRSTVO ZA GOSPODARSKI RAZVOJ IN TEHNOLOGIJO

ROS and C++

Simple Client

```
 #include "ros/ros.h"
#include "beginner_tutorials/AddTwoInts.h"
#include <cstdlib>

int main(int argc, char **argv)
{
    ros::init(argc, argv, "add_two_ints_client");
    if (argc != 3)
    {
        ROS_INFO("usage: add_two_ints_client X Y");
        return 1;
  }
```

Naložba v vašo prihodnost
Operacijo delno financira Evropska unija
Evropski sklad za regionalni razvoj
Investition in Ihre Zukunft
Operation teilfinanziert von der Europäischen Union
Europäischer Fonds für regionale Entwicklung
REPUBLIKA SLOVENIJA
MINISTRSTVO ZA GOSPODARSKI
RAZVOJ IN TEHNOLOGIJO

ROS and C++

Simple Client

```
ros::NodeHandle n;
 ros::ServiceClient client = n.serviceClient<beginner_tutorials::AddTwoInts>("add_two_ints");
 beginner_tutorials::AddTwoInts srv;
 srv.request.a = atoll(argv[1]);
 srv.request.b = atoll(argv[2]);
 if (client.call(srv))
 {
     ROS_INFO("Sum: %ld", (long int)srv.response.sum);
 }
 else
 {
     ROS_ERROR("Failed to call service add_two_ints");
     return 1;
 }

 return 0;
}
```

Naložba v vašo prihodnost
Operacijo delno financira Evropska unija
Evropski sklad za regionalni razvoj
Investition in Ihre Zukunft
Operation teilfinanziert von der Europäischen Union
Europäischer Fonds für regionale Entwicklung
REPUBLIKA SLOVENIJA
MINISTRSTVO ZA GOSPODARSKI
RAZVOJ IN TEHNOLOGIJO

ROS and C++

Simple Client

- Let's break the code down:

```
ros::ServiceClient client =
n.serviceClient<beginner_tutorials::AddTwoInts>("add_two_ints");
```

- This creates a client for the add_two_ints service. The ros::ServiceClient object is used to call the service later on.

```
beginner_tutorials::AddTwoInts srv;
srv.request.a = atoll(argv[1]);
srv.request.b = atoll(argv[2]);
```

- Here we instantiate an autogenerated service class, and assign values into its request member. A service class contains two members, request and response. It also contains two class definitions, Request and Response

Naložba v vašo prihodnost
Operacijo delno financira Evropska unija
Evropski sklad za regionalni razvoj
Investition in Ihre Zukunft
Operation teilfinanziert von der Europäischen Union
Europäischer Fonds für regionale Entwicklung
REPUBLIKA SLOVENIJA
MINISTRSTVO ZA GOSPODARSKI
RAZVOJ IN TEHNOLOGIJO

ROS and C++

Simple Client

if (client.call(srv))

- This actually calls the service. Since service calls are blocking, it will return once the call is done. If the service call succeeded, call() will return true and the value in srv.response will be valid. If the call did not succeed, call() will return false and the value in srv.response will be invalid.

- Let‘s build code.

Naložba v vašo prihodnost
Operacijo delno financira Evropska unija
Evropski sklad za regionalni razvoj
Investition in Ihre Zukunft
Operation teilfinanziert von der Europäischen Union
Europäischer Fonds für regionale Entwicklung
REPUBLIKA SLOVENIJA
MINISTRSTVO ZA GOSPODARSKI
RAZVOJ IN TEHNOLOGIJO

ROS and C++

Simple Client

- Go to your beginnger tutorials folder:

```
$roscd beginer_tutorial
```

- Open whit rosed or gedit CMakeList.txt and add following lines:

```
$gedit CMakeList.txt
```

```
rosbuild_add_executable(add_two_ints_server src/add_two_ints_server.cpp)
rosbuild_add_executable(add_two_ints_client src/add_two_ints_client.cpp)
```

- Build our package:

```
$rosmake beginner_tutorials
```

Naložba v vašo prihodnost
Operacijo delno financira Evropska unija
Evropski sklad za regionalni razvoj
Investition in Ihre Zukunft
Operation teilfinanziert von der Europäischen Union
Europäischer Fonds für regionale Entwicklung
REPUBLIKA SLOVENIJA
MINISTRSTVO ZA GOSPODARSKI
RAZVOJ IN TEHNOLOGIJO

ROS CONTENT

- PRESENTATION
- BASIC
- NODE
- TOPICS and MESSAGES
- ROS and C++ (Simple Publisher and Subscriber)
- SERVICES and PARAMETERS
- ROS and C++ (Simple Service and Client)
- TOOLS

Naložba v vašo prihodnost
Operacijo delno financira Evropska unija
Evropski sklad za regionalni razvoj
Investition in Ihre Zukunft
Operation teilfinanziert von der Europäischen Union
Europäischer Fonds für regionale Entwicklung
REPUBLIKA SLOVENIJA
MINISTRSTVO ZA GOSPODARSKI
RAZVOJ IN TEHNOLOGIJO

TOOLS

rqtconsole

- Tool rqt_console attaches to ROS's logging framework to display output from nodes. rqt_logger_level allows us to change the verbosity level (DEBUG, WARN, INFO, and ERROR) of nodes as they run:

```
$rosrun rqt_console rqt_console
```

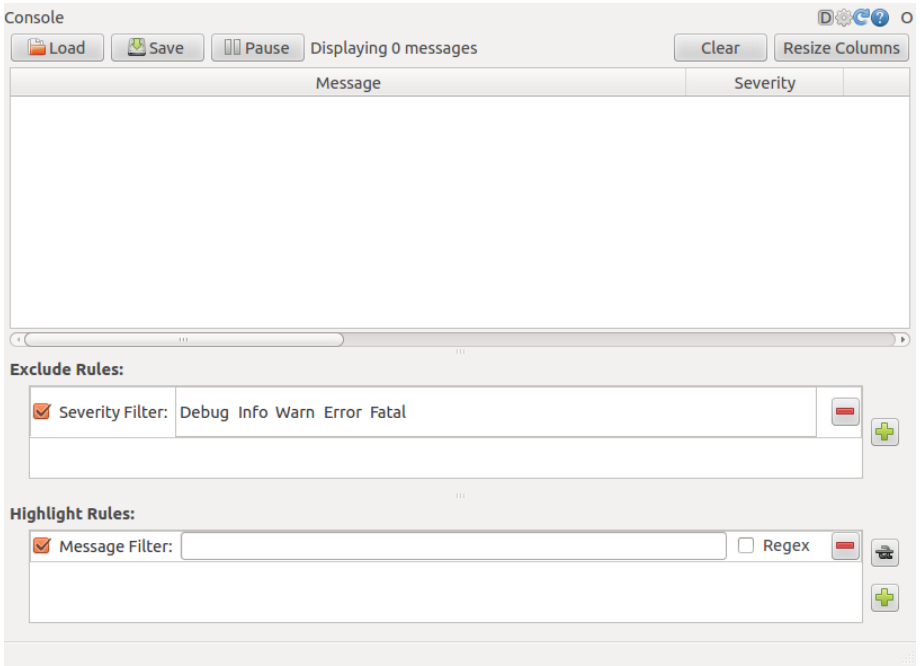
- And in new terminal:

```
$rosrun rqt_logger_level rqt_logger_level
```

Naložba v vašo prihodnost
Operacijo delno financira Evropska unija
Evropski sklad za regionalni razvoj
Investition in Ihre Zukunft
Operation teilfinanziert von der Europäischen Union
Europäischer Fonds für regionale Entwicklung
REPUBLIKA SLOVENIJA
MINISTRSTVO ZA GOSPODARSKI RAZVOJ IN TEHNOLOGIJO

TOOLS

rqtconsole



Naložba v vašo prihodnost
Operacijo delno financira Evropska unija
Evropski sklad za regionalni razvoj
Investition in Ihre Zukunft
Operation teilfinanziert von der Europäischen Union
Europäischer Fonds für regionale Entwicklung
REPUBLIKA SLOVENIJA
MINISTRSTVO ZA GOSPODARSKI
RAZVOJ IN TEHNOLOGIJO

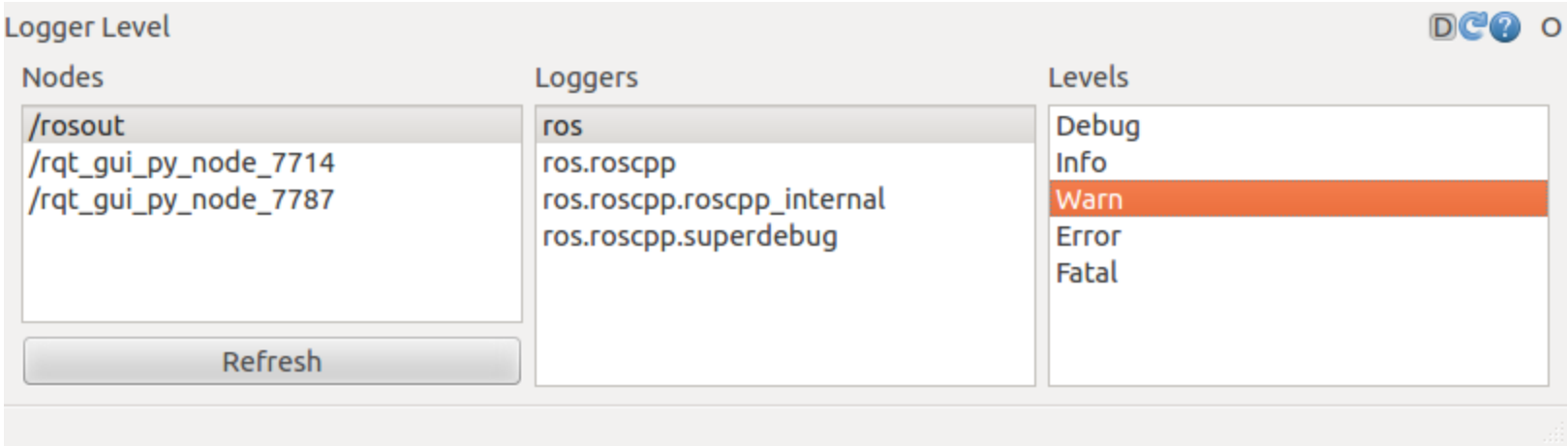
TOOLS

rqtconsole

- Now let's start turtlesim in a **new terminal**:

```
$rosrun turtlesim turtlesim_node
```

- Take a look what happens in console and rx_logger_level!
- Now let's change the logger level to Warn by refreshing the nodes in the rqt_logger_level window and selecting Warn as shown below.



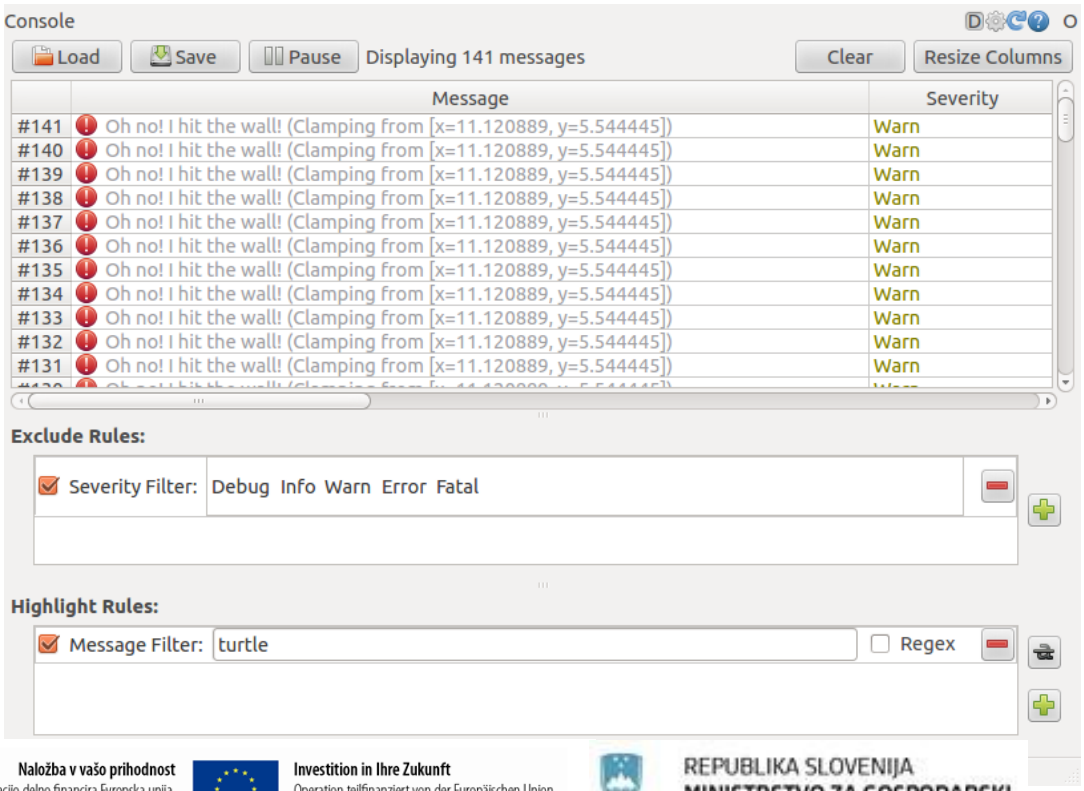
Naložba v vašo prihodnost
Operacijo delno financira Evropska unija
Evropski sklad za regionalni razvoj
Investition in Ihre Zukunft
Operation teilfinanziert von der Europäischen Union
Europäischer Fonds für regionale Entwicklung
REPUBLIKA SLOVENIJA
MINISTRSTVO ZA GOSPODARSKI
RAZVOJ IN TEHNOLOGIJO

TOOLS

rqtconsole

- Now let's run our turtle into the wall and see what is displayed in our rqt_console:

```
$rostopic pub /turtle1/command_velocity turtlesim/Velocity -r 1 -- 2.0 0.0
```



TOOLS

rqtconsole

- Logging levels are prioritized in the following order:

```
Fatal
Error
Warn
Info
Debug
```

- Fatal has the highest priority and Debug has the lowest. By setting the logger level, you will get all messages of that priority level or higher. For example, by setting the level to Warn, you will get all Warn, Error, and Fatal logging messages.
- Let's Ctrl-C our turtlesim and let's use roslaunch to bring up multiple turtlesim nodes and a mimicking node to cause one turtlesim to mimic another:

TOOLS

roslaunch

- Command roslaunch starts nodes as defined in a launch file.
- Usage:

```
$roslaunch [package] [filename.launch]
```

- First go to the beginner_tutorials package:

```
$roscd beginner_tutorials
```

- Then let's make a launch directory:

```
$mkdir launch
$cd launch
```

Naložba v vašo prihodnost
Operacijo delno financira Evropska unija
Evropski sklad za regionalni razvoj
Investition in Ihre Zukunft
Operation teilfinanziert von der Europäischen Union
Europäischer Fonds für regionale Entwicklung
REPUBLIKA SLOVENIJA
MINISTRSTVO ZA GOSPODARSKI
RAZVOJ IN TEHNOLOGIJO

TOOLS

roslaunch

- Paste inside following:

```
<launch>
<group ns="turtlesim1">
        <node pkg="turtlesim" name="sim" type="turtlesim_node"/>
</group>

<group ns="turtlesim2">
        <node pkg="turtlesim" name="sim" type="turtlesim_node"/>
</group>

<node pkg="turtlesim" name="mimic" type="mimic">
        <remap from="input" to="turtlesim1/turtle1"/>
        <remap from="output" to="turtlesim2/turtle1"/>
</node>
</launch>
```

Naložba v vašo prihodnost
Operacijo delno financira Evropska unija
Evropski sklad za regionalni razvoj
Investition in Ihre Zukunft
Operation teilfinanziert von der Europäischen Union
Europäischer Fonds für regionale Entwicklung
REPUBLIKA SLOVENIJA
MINISTRSTVO ZA GOSPODARSKI RAZVOJ IN TEHNOLOGIJO

TOOLS

roslaunch

- Take a look at the code:

```
<launch>
```

- Here we start the launch file with the launch tag, so that the file is identified as a launch file.

```
<group ns="turtlesim1">
<node pkg="turtlesim" name="sim" type="turtlesim_node"/>
</group>
<group ns="turtlesim2">
<node pkg="turtlesim" name="sim" type="turtlesim_node"/>
</group>
```

- Here we start two groups with a namespace tag of turtlesim1 and turtlesim2 with a turtlesim node with a name of sim. This allows us to start two simulators without having name conflicts

TOOLS

roslaunch

```
<node pkg="turtlesim" name="mimic" type="mimic">
<remap from="input" to="turtlesim1/turtle1"/>
<remap from="output" to="turtlesim2/turtle1"/>
</node>
```

- Here we start the mimic node with the topics input and output renamed to turtlesim1 and turtlesim2. This renaming will cause turtlesim2 to mimic turtlesim1.

```
</launch>
```

- This closes the xml tag for the launch file.

Naložba v vašo prihodnost
Operacijo delno financira Evropska unija
Evropski sklad za regionalni razvoj
Investition in Ihre Zukunft
Operation teilfinanziert von der Europäischen Union
Europäischer Fonds für regionale Entwicklung
REPUBLIKA SLOVENIJA
MINISTRSTVO ZA GOSPODARSKI
RAZVOJ IN TEHNOLOGIJO

TOOLS

roslaunch

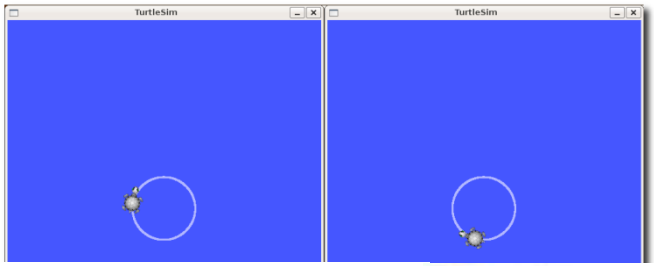
- Now let's roslaunch the launch file:

```
$roslaunch beginner_tutorials turtlemimic.launch
```

- Two turtlesims will start and in a **new terminal** send the rostopic command:

```
$rostopic pub /turtlesim1/turtle1/command_velocity turtlesim/Velocity -r 1 -- 2.0 -1.8
```

- You will see the two turtlesims start moving even though the publish command is only being sent to turtlesim1:



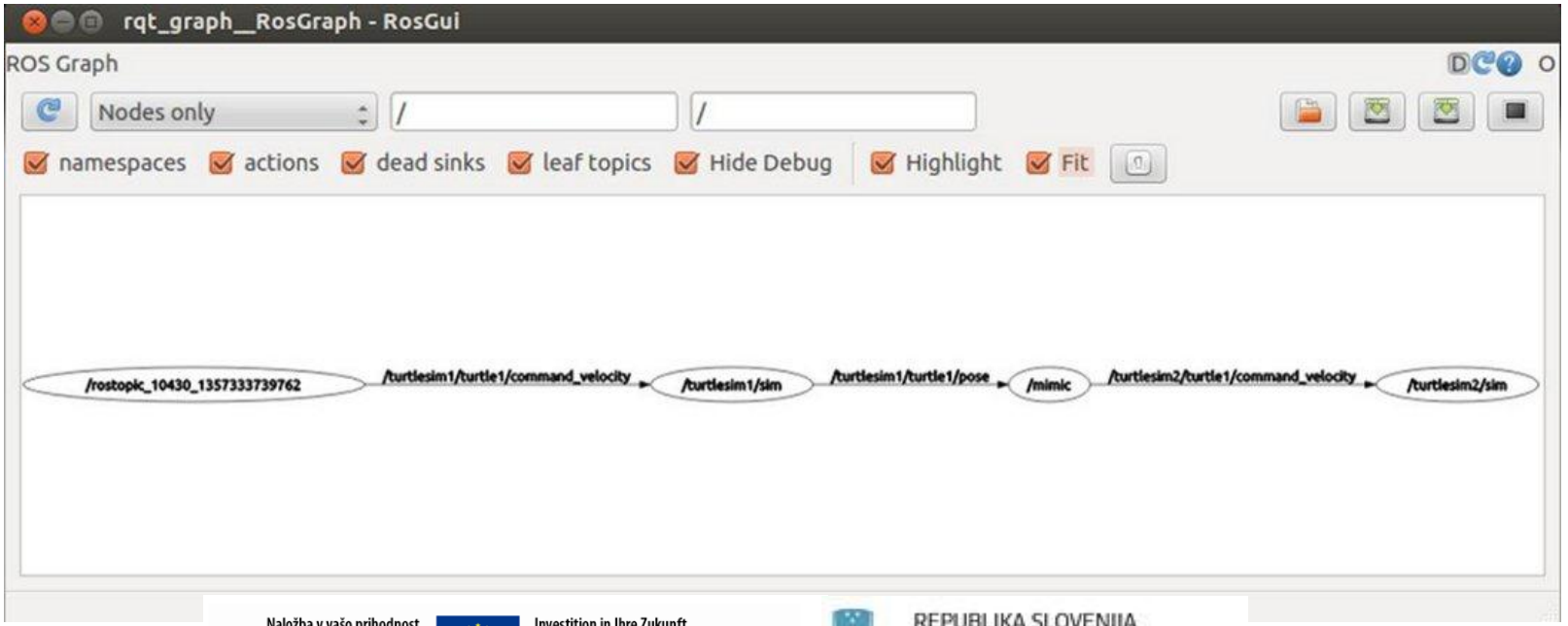
Naložba v vašo prihodnost
Operacijo delno financira Evropska unija
Evropski sklad za regionalni razvoj
Investition in Ihre Zukunft
Operation teilfinanziert von der Europäischen Union
Europäischer Fonds für regionale Entwicklung
REPUBLIKA SLOVENIJA
MINISTRSTVO ZA GOSPODARSKI
RAZVOJ IN TEHNOLOGIJO

TOOLS

roslaunch

- We can also use rqt_graph to better understand what our launch file did. Run rqt's main window and select rqt_graph:

```
$rqt
$rgt_graph
```



TOOLS

rosbag

- This section of the tutorial will instruct you how to record topic data from a running ROS system. The topic data will be accumulated in a bag file.
- First, execute the following two commands in newterminals:

```
$roscore
$rosrun turtlesim turtlesim_node
$rosrun turtlesim turtlesim_teleop_key
```

- This will start two nodes - the turtlesim visualizer and a node that allows for the keyboard control of turtlesim using the arrows keys on the keyboard. If you select the terminal window from which you launched turtle_keyboard, you should see something like the following:

```
Reading from keyboard
---------------------------
Use arrow keys to move the turtle.
```

Naložba v vašo prihodnost
Operacijo delno financira Evropska unija
Evropski sklad za regionalni razvoj
Investition in Ihre Zukunft
Operation teilfinanziert von der Europäischen Union
Europäischer Fonds für regionale Entwicklung
REPUBLIKA SLOVENIJA
MINISTRSTVO ZA GOSPODARSKI
RAZVOJ IN TEHNOLOGIJO

TOOLS

rosbag

- List avalible topics:

```
$rostopic list –v
```

```
Published topics:
 * /turtle1/color_sensor [turtlesim/Color] 1 publisher
 * /turtle1/command_velocity [turtlesim/Velocity] 1 publisher
 * /rosout [roslib/Log] 2 publishers
 * /rosout_agg [roslib/Log] 1 publisher
 * /turtle1/pose [turtlesim/Pose] 1 publisher

Subscribed topics:
 * /turtle1/command_velocity [turtlesim/Velocity] 1 subscriber
 * /rosout [roslib/Log] 1 subscriber
```

- Make new directory where you will record data:

```
$mkdir ~/bagfiles
$cd ~/bagfiles
$rosbag record -a
```

Naložba v vašo prihodnost
Operacijo delno financira Evropska unija
Evropski sklad za regionalni razvoj
Investition in Ihre Zukunft
Operation teilfinanziert von der Europäischen Union
Europäischer Fonds für regionale Entwicklung
REPUBLIKA SLOVENIJA
MINISTRSTVO ZA GOSPODARSKI RAZVOJ IN TEHNOLOGIJO

TOOLS

rosbag

- We have now recorder all topics, you can record also separete topic:

```
$rosbag record [topic1] [topic2] …
```

- Now go in teleop_key terminal and move turtle around for few seconds:

- To stop recording press Ctrl+C

- Play your bag file:

```
$rosbag play –l <your_bag_file>
```

- Plays your bag in a loop.

Naložba v vašo prihodnost
Operacijo delno financira Evropska unija
Evropski sklad za regionalni razvoj
Investition in Ihre Zukunft
Operation teilfinanziert von der Europäischen Union
Europäischer Fonds für regionale Entwicklung
REPUBLIKA SLOVENIJA
MINISTRSTVO ZA GOSPODARSKI
RAZVOJ IN TEHNOLOGIJO

TOOLS

roswtf

- Make sure that roscore **is not running!**
- Command roswtf examines your system to try and find problems.
- Let's try it out:

```
$roscd
$roswtf
```

```
Stack: ros
================================================================================
Static checks summary:

No errors or warnings
================================================================================

Cannot communicate with master, ignoring graph checks
```

- Now run you roscore, and try again:

```
$roscore
$roscd
$roswtf
```

Naložba v vašo prihodnost
Operacijo delno financira Evropska unija
Evropski sklad za regionalni razvoj
Investition in Ihre Zukunft
Operation teilfinanziert von der Europäischen Union
Europäischer Fonds für regionale Entwicklung
REPUBLIKA SLOVENIJA
MINISTRSTVO ZA GOSPODARSKI
RAZVOJ IN TEHNOLOGIJO

TOOLS

roswtf

- Command roswtf did some online examination of your graph now that your roscore is running. Depending on how many ROS nodes you have running, this can take a long time to complete. As you can see, this time it produced a warning.

```
WARNING The following node subscriptions are unconnected:
 * /rosout:
   * /rosout
```

- Command roswtf is warning you that the rosout node is subscribed to a topic that no one is publishing to. In this case, this is expected because nothing else is running, so we can ignore it.
- Command roswtf will warn you about things that look suspicious but may be normal in your system. It can also report errors for problems that it knows are wrong.

TOOLS

Summary

`$rqt_console:`	console for outputs
`$rqt_logger_lever:`	you can change priority level of outputs
`$roslaunch:`	start multiple nodes
`$rqt_graph:`	graph of nodes and their connections
`$rosbag record:`	record data to file
`$rosbag play:`	play recorded file
`$roswtf:`	shows problem in ros system

Naložba v vašo prihodnost
Operacijo delno financira Evropska unija
Evropski sklad za regionalni razvoj
Investition in Ihre Zukunft
Operation teilfinanziert von der Europäischen Union
Europäischer Fonds für regionale Entwicklung
REPUBLIKA SLOVENIJA
MINISTRSTVO ZA GOSPODARSKI
RAZVOJ IN TEHNOLOGIJO

ROS CONTENT

- PRESENTATION
- BASIC
- NODE
- TOPICS and MESSAGES
- ROS and C++ (Simple Publisher and Subscriber)
- SERVICES and PARAMETERS
- ROS and C++ (Simple Service and Client)
- TOOLS

Naložba v vašo prihodnost
Operacijo delno financira Evropska unija
Evropski sklad za regionalni razvoj
Investition in Ihre Zukunft
Operation teilfinanziert von der Europäischen Union
Europäischer Fonds für regionale Entwicklung
REPUBLIKA SLOVENIJA
MINISTRSTVO ZA GOSPODARSKI
RAZVOJ IN TEHNOLOGIJO

What next

- Answer questionare on ROS:

  Find link in: http://wiki.ros.org/ROS/Tutorials

- Avalible video tutorials: http://wiki.ros.org/ROS/Tutorials

- Using simulation of robot model in Gazebo – tutorial

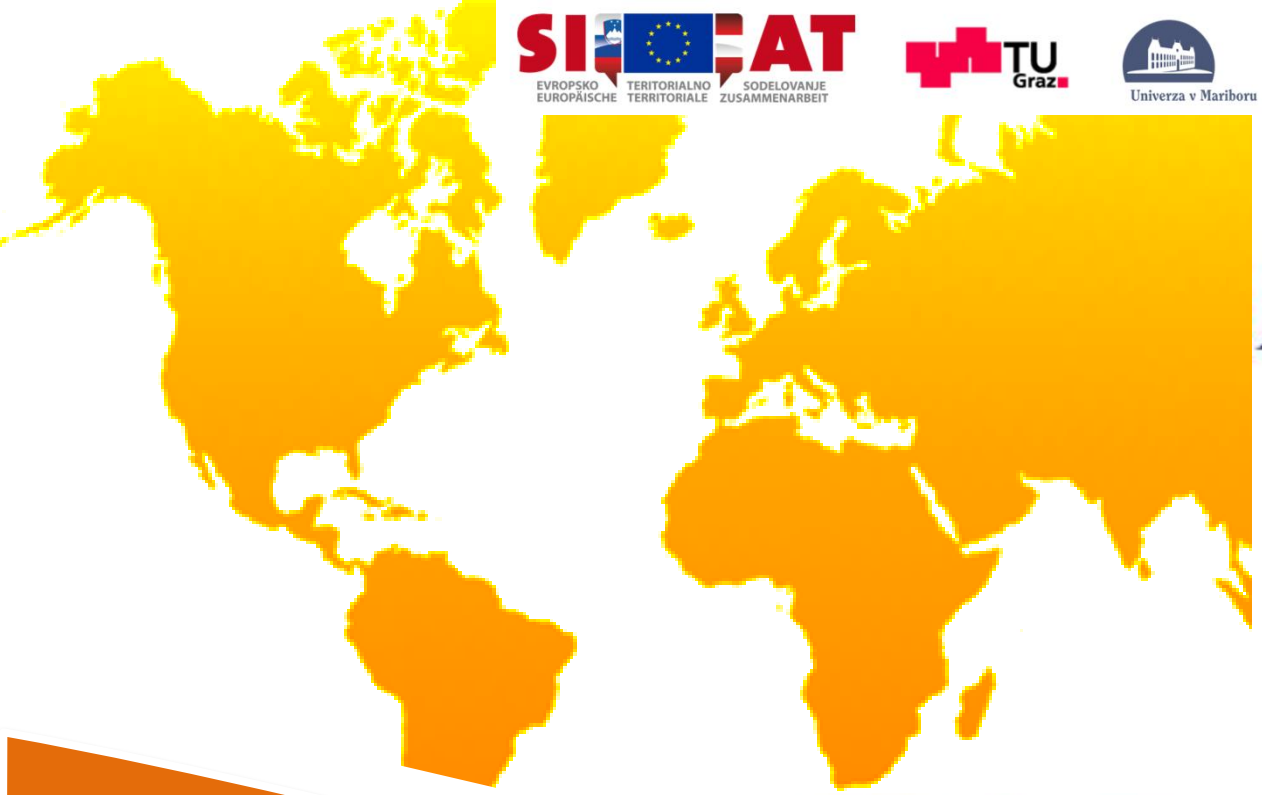
- Practical Sessions

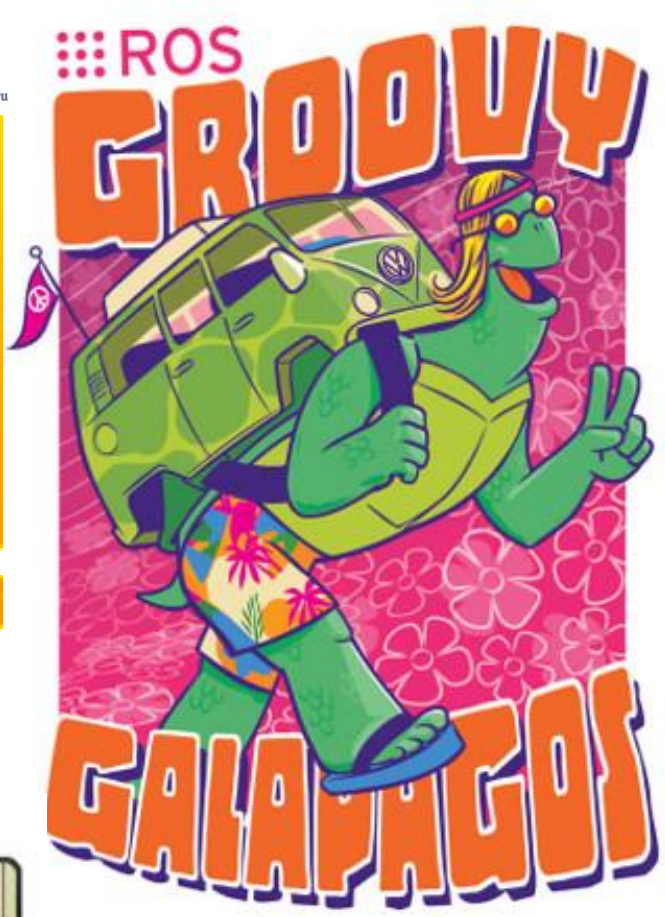
Naložba v vašo prihodnost
Operacijo delno financira Evropska unija
Evropski sklad za regionalni razvoj
Investition in Ihre Zukunft
Operation teilfinanziert von der Europäischen Union
Europäischer Fonds für regionale Entwicklung
REPUBLIKA SLOVENIJA
MINISTRSTVO ZA GOSPODARSKI
RAZVOJ IN TEHNOLOGIJO

What next

- Form teams or individual:
- Practical Sessions: Proposed themes
  - Gazebo: erratic simulator
  - Stage simulator
  - Use kinect and openni_tracker
  - Use kinect and rgbdslam
  - Robotis Servo motors
  - Laser Range Scanner and mapping
  - Navigation stack
  - Mapping with Laser Scanner
  - Exploration
  - Your own idea…
- Final presentation of your work on Friday!
- Presentations are avalible at: http://www.tedusar.eu/cms/sl/summerschool2013

Naložba v vašo prihodnost
Operacijo delno financira Evropska unija
Evropski sklad za regionalni razvoj
Investition in Ihre Zukunft
Operation teilfinanziert von der Europäischen Union
Europäischer Fonds für regionale Entwicklung
REPUBLIKA SLOVENIJA
MINISTRSTVO ZA GOSPODARSKI
RAZVOJ IN TEHNOLOGIJO





ROS tutorial

Peter Lepej
peter.lepej@uni-mb.si
Lab: G2.2N.10 Tesla
Tel: (02) 220 7336

www.tedusar.eu
www.si-at.eu

Naložba v vašo prihodnost
Operacijo delno financira Evropska unija
Evropski sklad za regionalni razvoj

Investition in Ihre Zukunft
Operation teilfinanziert von der Europäischen Union
Europäischer Fonds für regionale Entwicklung

REPUBLIKA SLOVENIJA
MINISTRSTVO ZA GOSPODARSKI
RAZVOJ IN TEHNOLOGIJO