



[Virtual Robotic Laboratory and Learning Materials for ROSin](#) by [Inovasyon Muhendislik Ltd. Sti.](#) is licensed under [CC BY-NC-ND 4.0](#)

ROS Uygulamalı Eğitimleri – 2019

Robot Operating System (ROS) ve Uygulamaları



Supported by ROSIN - ROS-Industrial Quality-Assured Robot Software Components.
More information: rosin-project.eu



This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No. 732287.

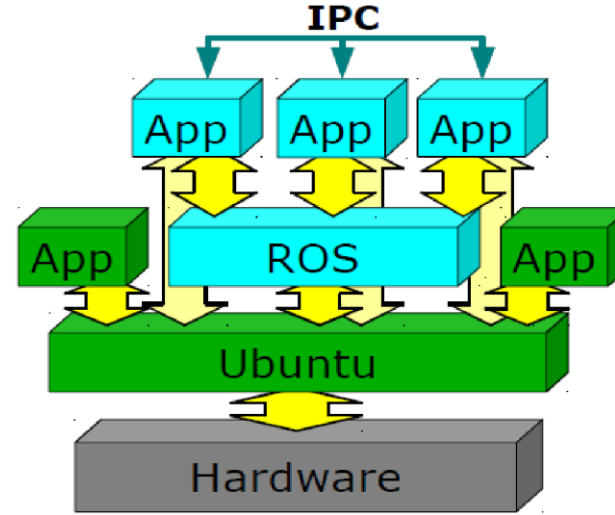
Annex_4_2_ROS_Training_tr.pptx

İçerik – 09.00-09.45

- ROS Nedir – Ne Değildir?
- Neden ROS Kullanmalıyız?
- ROS Sensörler ve ROS Kullanan Firmalar
- ROS Kurulumu
- Linux Temel Kodlar

ROS Nedir – Ne Değildir?

- Programlama dili
- Kütüphane
- İşletim sistemi
- Entegre geliştirme ortamı
- Bünyesinde servisler ve çeşitli makroları çalıştıran, açık kaynaklı robotlar için meta işletim sistemidir.



Neden ROS Kullanmalıyız?

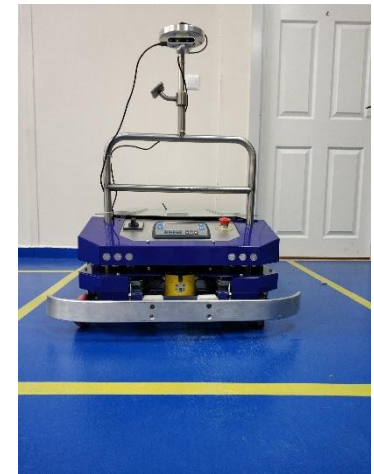
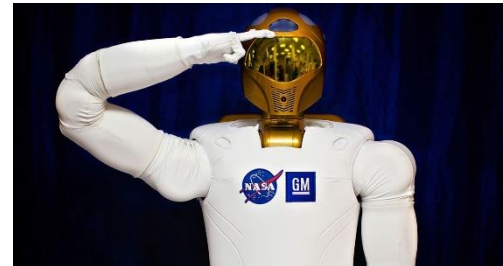
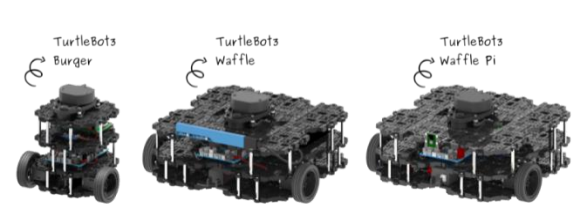


- Dil bağımsız yapı (C++, Python, Lisp, Java, Lua)
- Modüler çalışma, parametreler, mesajlar ve servisler sayesinde programlara ani müdahale imkanı
- Node/topic ile kolay ve sistematik veri aktarımı
- Birçok sensör, motor ve robot platformu için sürücü desteği
- Açık kaynak kodlu
- Haritalama, konumlandırma ve algılama amaçlı hazır algoritma, kütüphane ve paketler
- Aktif topluluk
- Hızlı test etme
- Donanım soyutlaması
- Görselleştiriciler

ROS Sensörler



ROS Tercih Eden Firmalar



ROS Kurulumu – 1/2

Supported:



Ubuntu Wily amd64 i386
Xenial amd64 i386 armhf arm64

[Source installation](#)

Experimental:



OS X (Homebrew)



Gentoo



OpenEmbedded/Yocto



Debian Jessie amd64 arm64

Unofficial Installation Alternatives:



Single line install A single line command to install ROS Kinetic on Ubuntu

ROS Melodic Morenia (Recommended)	May 23rd, 2018			May, 2023 (Bionic EOL)
ROS Lunar Loggerhead	May 23rd, 2017			May, 2019
ROS Kinetic Kame	May 23rd, 2016			April, 2021 (Xenial EOL)
ROS Jade Turtle	May 23rd, 2015			May, 2017
ROS Indigo Igloo	July 22nd, 2014			April, 2019 (Trusty EOL)
ROS Hydro Medusa	September 4th, 2013			May, 2015
ROS Groovy Galapagos	December 31, 2012			July, 2014
ROS Fuerte Turtle	April 23, 2012			—
ROS Electric Emys	August 30, 2011			—
ROS Diamondback	March 2, 2011			—
ROS C Turtle	August 2, 2010			—
ROS Box Turtle	March 2, 2010			—

ROS Kurulumu – 2/2

1. Bilgisayarın, settings.ros.org adresinden yazılımı kabul edecek şekilde ayarlanması

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

2. Anahtarları ayarlama

```
sudo apt-key adv --keyserver
'hkp://keyserver.ubuntu.com:80' --recv- key
C1CF6E31E6BADE8868B172B4F42ED6FBAB17C654
```

3. Debian paketinin güncelliğinden emin olma

```
sudo apt-get update
```

4. Tam sürüm kurulum yapma
(ROS,rqt,rviz,robot-generic kütüphaneler,
2D/3D simülatörler ...)

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

5. *rosdep* in başlatılması ve güncellenmesi

```
sudo rosdep init
```

```
rosdep update
```

6. Ortam kurulumu

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

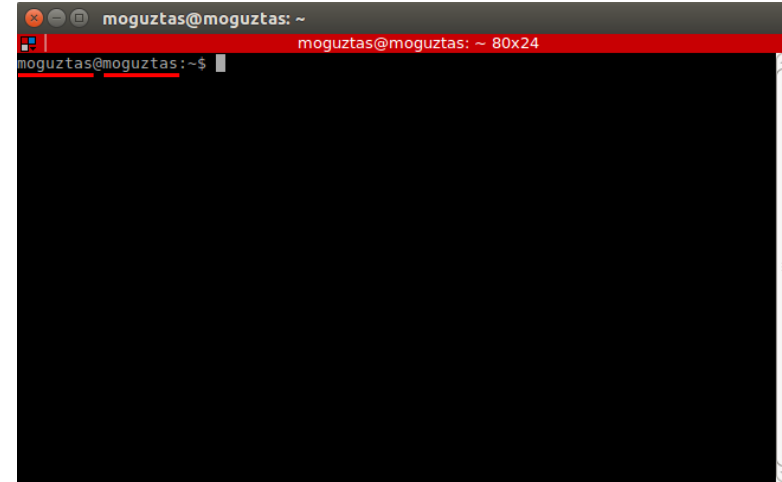
```
source ~/.bashrc
```

7. Derleme paketleri için bağımlılıkların yüklenmesi

```
sudo apt install python-rosinstall python-rosinstall-
generator python-wstool build-essential
```


Linux Temel Kodlar – 1/4

- Yeni terminal açma: Ctrl + Alt + T
- moguztas@moguztas:
 - moguztas: kullanıcı adı
 - moguztas: bilgisayar adı
- Kopyalama: Ctrl + Shift + C
- Yapıştır: Ctrl + Shift + V
- İşlemi Durdurma: Ctrl + C



Linux Temel Kodlar – 2/4

- **cd** : Change Directory anlamına gelir. Belirtilen yoldaki klasöre giriş yapılır.
- **ls** : List anlamına gelir. Girilen klasördeki dosya ve dökümanları listeler.
- **mkdir <klasor_adi>**: Make Directory anlamına gelir. Bulunulan yolda klasör oluşturur.
- **mkdir -p <klasore_giden_yol>/<klasor_adi>**: Klasör oluşturma işlemini tüm yol üzerinde gerçekler. Eğer olmayan klasörler varsa iç içe o klasörleri oluşturur.
- **rm <silinecek_dosya>**: Remove anlamına gelir. Bir dosyayı silmek için kullanılır.
- **rm -rf <silinecek_dosya>**: Recursive Remove anlamına gelir. Birden fazla dosyayı silmek için kullanılır.
- **mv <tasinacak_dosya> <tasinacak_klasor>**: Dosya taşıma işlemi yapar. Rename işlemi de mv komutu kullanılarak yapılabilir.
- **cp <kopyalanacak_dosya> <kopyalanacak_klasor>**: Kopyalama işlemi yapar.
- **wget '<indirilecek_url>'** : Bilgisayarda bulunulan klasöre, internet üzerinde belirtilen dosyayı indirir.

Linux Temel Kodlar – 3/4

- **sudo apt-get install <paket_ismi>**: Repository'lerde paketi arar ve bilgisayara kurar.
- **sudo apt-get remove <paket_ismi>**: Bilgisayarda paketi arar ve bilgisayardan siler.
- **sudo apt-get update**: sources.list dosyasında kayıtlı olan repositorylerin bilgisini bilgisayara alır.
- **sudo apt-get upgrade**: Bilgisayarda bulunan paketlerin güncellemesini yapar.
- **apt-cache search <paket_ismi>**: İlgili paketi repositorylerde arar ve getirir.
- **sudo chmod <izin_tipi> <klasor_veya_dosya_ismi>**: İlgili klasör veya dökümana dosya izinleri verir. İzin tipi olarak 777 kullanılması, Read-Write-Execute anlamına gelir.
- **sudo su**: Super user olarak işlemlerimizi yapmamızı sağlar.
- **sudo service <servis_ismi> start** : Linux bünyesinde çalışan servisi başlatır.
- **sudo service <servis_ismi> stop** : Linux bünyesinde çalışan servisi sonlandırır.
- **sudo service <servis_ismi> restart** : Linux bünyesinde çalışan servisi yeniden başlatır.

Linux Temel Kodlar – 4/4

- **history**: Terminaldeki kod geçmişini gösterir.
- **clear**: Terminaldeki kodları siler.
- **ps** : Processes anlamına gelir. Linux sistem üzerinde çalışan işlemleri listeler.
- **ps -aux | grep <işlem_ismi>**: Linux üzerinde çalışan belirli işlemi veya işlemleri getirir.
- **kill -9 <işlem_ID'si>** : Linux üzerinde çalışan ve üstteki komut ile ID'si bulunan işlemi öldürmeyi sağlar.
- **udo lsusb** : Linux sistem üzerinde kayıtlı ve çalışan USB cihazlarını listeler.
- **cat** : Bir dosyanın içeriğini terminal ekranına bastırır.
- **pwd** : Bir dosyanın yolunu terminal ekranına bastırır.
- **echo <değişken>** : Linux terminali üzerinde kayıtlı global değişkenlerin ve sonradan tanımlanan değişkenlerin ekrana bastırılmasını sağlar.
- **<editör_ismi> <dosya>** : Linux'da bir dosyayı düzenlemeyi sağlar.
- **setxkbmap tr** : Klavyeyi türkçe karakterli hale getirir.
- **g++ hello_world.cpp -o hello_world** : GNU C Compiler aracılığı ile C/C++ dosyalarını derler.
- **./hello_world** : Executable dosya çalıştırma

İçerik – 10.00-11.45

- ROS Mimarisi Giriş
- Dosya Sistemi – File System Level
 - Paketler (Packages)
 - Metapaketler (Metapackages)
 - Paket bildirileri (Package Manifests)
 - Mesaj tipleri (Message types)
 - Servis tipleri (Service types)
- İşlem Grafiği – Computation Graph Level
 - Düğümler (Nodes)
 - Parametre Servisi (Parameter Service)
 - Mesajlar (Messages)
 - Konular (Topics)
 - Servisler (Services)
 - Çantalar (Bags)
 - Ana (Master)
- Topluluk – Community Level
- Publisher-Subscriber ve Service Client Yapıları
- ROS Araçları

ROS Mimarisi

ROS mimarisi 3 bölüme ayrılmıştır:

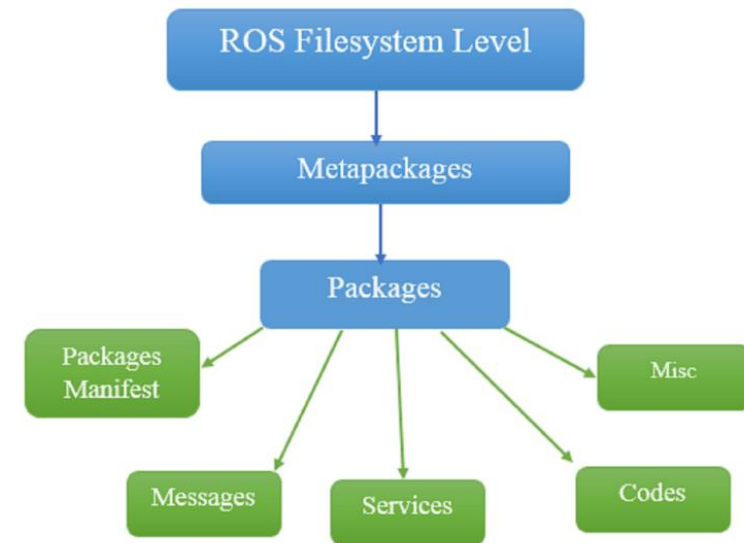
Dosya Sistemi (Filesystem Level): ROS'un dahili olarak nasıl oluşturulduğunu, klasör yapısını ve çalışması gereken minimum dosya sayısını açıklamak için bir grup kavramı içerir.

İşlem Grafiği (Computation Graph Level): ROS'un işlemler ve sistemler arasındaki iletişimi nasıl kullandığını açıklayan kavramları içerir.

Topluluk (Community Level): Geliştiriciler arasında bilgi, algoritma ve kod paylaşmaya yönelik bir dizi araç ve kavram içerir.

Dosya Sistemi – Filesystem Level

- Paketler (Packages)
- Metapaketler (Metapackages)
- Paket bildirileri (Package Manifests)
- Mesaj tipleri (Message types)
- Servis tipleri (Service types)



Paketler (Packages) – 1/3

- Bir paket, işlemleri (düğümleri), ROS'a bağlı kütüphaneleri, veri kümelerini, yapılandırma dosyalarını veya faydalı herhangi bir şeyi içerebilir.
- Paketlerin oluşturulma amacı kodları ufak parçalara bölerek yeniden kullanılabilir olmasını sağlamaktır.
- Paketler, ROS'ta yazılımın düzenli olmasını sağlayan ana birimlerdir.
- Paketler genellikle tipik dosya ve klasörlerden oluşur:
 - **include/paket_adı/**: Bu dizinde, ihtiyaç duyduğumuz kütüphanelerin (libraries) başlık (header) dosyaları bulunur.
 - **msg/**: Standart olarak bulunan mesaj tiplerinden farklı bir mesaj tipi oluşturulacaksa bu klasörün içinde oluşturulmalıdır.
 - **scripts/**: Python gibi direk çalıştırılabilir betik (script) dilleri ile yazılan kodlar bu klasör içinde yer almalıdır.
 - **src/**: Programların kaynak dosyalarının (source files) bulunduğu yerdir.
 - **srv/**: Servis (service) dosyalarının bulunduğu yerdir.
 - **CMakeLists.txt**: Derleyiciye verilen emirleri ve daha birçok yapı bilgisini içeren CMake derleme (built) dosyasıdır.
 - **package.xml**: Paketlerin bildiri dosyasıdır.

Paketler (Packages) – 2/3

ROS paketler oluşturmamıza, düzenlememize ve paketler ile çalışmamıza yardımcı olabilecek araçlara sahiptir:

- **rospack:** Paketleri bulma ve paketler hakkında bilgi edinmeye yarar.
- **roscrcatkin:** Yeni paket oluşturmaya yarar.
- **rosmake:** Paketleri derlemeye (compile) yarar.
- **rosdep:** Paketlerin bağımlılıklarını (dependencies) yüklemeye yarar.
- **catkin_create_pkg:** Yeni paket oluşturmaya yarar.

ROS'un paketler ile paketlerin klasörleri ve dosyaları arasında dolaşmamızı ve taşıma yapmamızı sağlayan *roscd* isimli bir paketi vardır. *roscd* aracında desteklenen bazı komutlar:

- **roscd:** ROS dizinleri arasında dolaşma imkanı sağlar.
- **roscp:** Dosyaları düzenlemeyi sağlar.
- **roscp:** Başka paketteki dosyaları kopyalamayı sağlar.
- **roscp:** Çalıştırılabilir (executable) dosyaların çalıştırılmasını sağlar.
- **rosls:** Paketteki dosyaları listelemeye yarar.

Paketler (Packages) – 3/3

Paket mutlaka *CMakeLists.txt* dosyasını içermelidir. Bu dosya *catkin* e kodların nasıl ve nereye yükleneceğini bildirir.

CMakeLists.txt dosyası aşağıdaki formatı izlemelidir, aksi halde paketler doğru şekilde oluşturulamaz.

1. **Required CMake Version** (cmake_minimum_required)
2. **Package Name** (project())
3. **Find other CMake/Catkin packages needed for build** (find_package())
4. **Enable Python module support** (catkin_python_setup())
5. **Message/Service/Action Generators** (add_message_files(), add_service_files(), add_action_files())
6. **Invoke message/service/action generation** (generate_messages())
7. **Specify package build info export** (catkin_package())
8. **Libraries/Executables to build** (add_library()/add_executable()/target_link_libraries())
9. **Tests to build** (catkin_add_gtest())
10. **Install rules** (install())

Satır numaralandırmayı aç/kapa

```
1  # Get the information about this package's buildtime dependencies
2  find_package(catkin REQUIRED
3      COMPONENTS message_generation std_msgs sensor_msgs)
4
5  # Declare the message files to be built
6  add_message_files(FILES
7      MyMessage1.msg
8      MyMessage2.msg
9  )
10
11 # Declare the service files to be built
12 add_service_files(FILES
13     MyService.srv
14 )
15
16 # Actually generate the language-specific message and service files
17 generate_messages(DEPENDENCIES std_msgs sensor_msgs)
18
19 # Declare that this catkin package's runtime dependencies
20 catkin_package(
21     CATKIN_DEPENDS message_runtime std_msgs sensor_msgs
22 )
23
24 # define executable using MyMessage1 etc.
25 add_executable(message_program src/main.cpp)
26 add_dependencies(message_program ${PROJECT_NAME}_EXPORTED_TARGETS) ${catkin_EXPORTED_T
27     ARGETS})
28
29 # define executable not using any messages/services provided by this package
30 add_executable(does_not_use_local_messages_program src/main.cpp)
31 add_dependencies(does_not_use_local_messages_program ${catkin_EXPORTED_TARGETS})
```

Metapaketler (Metapackages)

- Metapaketler paketleri organize olarak çalıştırmaya yarar (çoklu paket kümesini basitçe gruplandırır).
- Meta paketler, ROS'ta yalnızca *package.xml* dosyası içeren özel paketlerdir.

Örnek: *robot* metapaketinin içerdiği paketler: [control_msgs, diagnostics, executive_smach, filters, geometry, joint_state_publisher, kdl_parser, kdl_parser_py, robot_state_publisher, urdf, urdf_parser_plugin, xacro]

```
sudo apt-get install ros-$distro-robot
```

```
sudo apt-get install ros-$distro-actionlib ros-$distro-angles ros-$distro-bond_core ros-$distro-catkin ros-$distro-class_loader ros-$distro-cmake_modules ros-$distro-common_msgs ros-$distro-console_bridge ros-$distro-control_msgs ros-$distro-diagnostics ros-$distro-dynamic_reconfigure ros-$distro-executive_smach ros-$distro-filters ros-$distro-gencpp ros-$distro-geneus ros-$distro-genlisp ros-$distro-genmsg ros-$distro-gennodejs ros-$distro-genpy ros-$distro-geometry ros-$distro-message_generation ros-$distro-message_runtime ros-$distro-nodelet_core ros-$distro-pluginlib ros-$distro-robot_model ros-$distro-robot_state_publisher ros-$distro-ros ros-$distro-ros_comm ros-$distro-roscpp_core ros-$distro-roscpp_migration_rule ros-$distro-rosconsole_bridge ros-$distro-roscpp_core ros-$distro-rosgraph_msgs ros-$distro-roslisp ros-$distro-rospack ros-$distro-std_msgs ros-$distro-std_srvs ros-$distro-xacro
```

Paket Bildirileri (Package Manifests)

Paket bildirileri (**package.xml**), bir paket hakkında: adı, sürümü, açıklaması, lisans bilgileri, bağımlılıkları ve dışa aktarılan paketler gibi diğer bilgileri içeren bir dosyadır. Bu dosyanın oluşturulmasının sebebi paket yüklenmesinin ve dağıtımının kolaylaştırılmasıdır.

```
<package format="2">
  <name>foo_core</name>
  <version>1.2.4</version>
  <description>
    This package provides foo capability.
  </description>
  <maintainer email="ivana@willowgarage.com">Ivana Bildbotz</maintainer>
  <license>BSD</license>

  <url>http://ros.org/wiki/foo_core</url>
  <author>Ivana Bildbotz</author>

  <buildtool_depend>catkin</buildtool_depend>

  <depend>roscpp</depend>
  <depend>std_msgs</depend>

  <build_depend>message_generation</build_depend>

  <exec_depend>message_runtime</exec_depend>
  <exec_depend>rospy</exec_depend>

  <test_depend>python-mock</test_depend>

  <doc_depend>doxygen</doc_depend>
</package>
```

Paketin adı

Paketin sürüm numarası (3 noktayla ayrılmış tam sayı olması gerekir)

Paket içeriğinin açıklaması

Paketi sürdüren kişiler

Kodun yayınlandığı yazılım lisansları (ör. GPL, BSD, ASL).

Build Tool Dependencies, bu paketin kendisini oluşturmak için ihtiyaç duyduğu derleme sistemi araçlarını belirtir.

Depend, Paketin bağlı olduğu paketleri belirtir.

Build Dependencies, bu paketi oluşturmak için hangi paketlerin gerekli olduğunu belirtir.

Execution Dependencies, bu pakette kod çalıştırmak için hangi paketlerin gerekli olduğunu belirtir.

Test Dependencies, ünite testleri için ek bağımlılıklar belirler.

Documentation Tool Dependencies, bu paketin dokümantasyon oluşturmak için ihtiyaç duyduğu dokümantasyon araçlarını belirtir.

Mesaj Tipleri (Message Types)

Mesaj dosyası **msg/** klasöründe ve **.msg** uzantısına sahip olmalıdır (my_package/msg/MyMessageType.msg).

[geometry_msgs/Pose Message](#)

File: `geometry_msgs/Pose.msg`

Raw Message Definition

```
# A representation of pose in free space, composed of position and orientation.  
Point position  
Quaternion orientation
```

Compact Message Definition

```
geometry_msgs/Point position  
geometry_msgs/Quaternion orientation
```

Servis Tipleri (Service Types)

Servis dosyası **srv/** klasöründe ve **.srv** uzantısına sahip olmalıdır (my_package/srv/MyServiceType.srv).

[turtlesim/Spawn Service](#)

File: `turtlesim/Spawn.srv`

Raw Message Definition

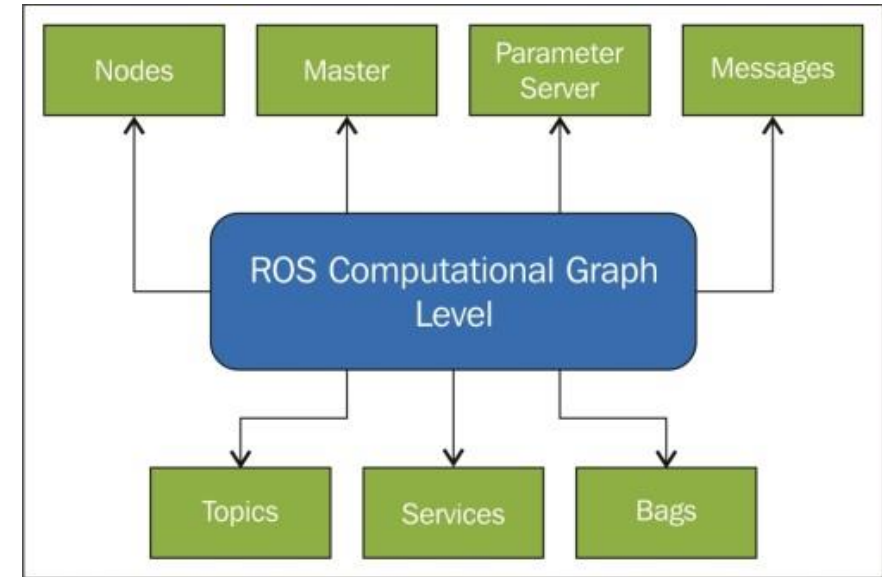
```
float32 x  
float32 y  
float32 theta  
string name # Optional. A unique name will be created and returned if this is empty  
---  
string name
```

Compact Message Definition

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

İşlem Grafiği – Computation Graph Level

- Düğümler (Nodes)
- Parametre Servisi (Parameter Service)
- Mesajlar (Messages)
- Konular (Topics)
- Servisler (Services)
- Çantalar (Bags)
- Ana (Master)



Düğüm (Nodes) – 1/2

- Düğüm, hesaplama yapan işlemlerdir.
- Bir düğüm, C++ için *roscpp* ve Python için *rospy* gibi farklı kütüphaneler kullanılarak yazılabilir.
- ROS'taki düğümleri kullanmak bize hata toleransı sağlar ve sistemi ve işlevleri basitleştirerek kodu ve işlevleri ayırır.
- Bir düğümün sistemde benzersiz bir adı olması gerekir.
- ROS düğümlerinin güçlü bir özelliği, düğümü başlatırken parametreleri (düğüm adı, konu adı, vb) değiştirebilme özelliğidir. Değiştirme işlemi ile, kod yeniden derlenmeden düğüm yapılandırılabilir, böylece farklı senaryolara kolayca adapte edilebilir.
 - Düğümdeki konu adını değiştirmeye örnek:
`roslaunch book_tutorials tutorialX topic1:=/level1/topic1`
 - Düğümdeki parametreleri değiştirmeye örnek:
`roslaunch book_tutorials tutorialX _param:=9.0`
- ROS, *nodelets* adında başka bir düğüm türüne sahiptir. Bu özel düğümler, tek bir işlemde birden fazla düğümü çalıştırmak üzere tasarlanmıştır. Bununla düğümler ağa aşırı yüklenmeden daha verimli iletişim kurabilirler. Nodelets özellikle aktarılan veri hacminin çok yüksek olduğu kamera sistemleri ve 3D sensörler için kullanışlıdır.

Not: Sistemde her şeyi yapan büyük bir düğüme sahip olmak yerine, yalnızca tek bir işlevsellik sağlayan birçok düğüme sahip olmak daha verimlidir.

Düğümmler (Nodes) – 2/2

ROS, düğümmleri işlemek ve bilgiler vermek için *roscall* aracına sahiptir. *roscall* aracında desteklenen bazı komutlar:

- **roscall info node_name**: Düğüm hakkındaki bilgileri yazdırır.
- **roscall kill node_name**: Çalışan bir düğümü sonlandırır.
- **roscall list**: Aktif düğümmleri listeler.
- **roscall machine hostname**: Belirli bir makinede çalışan düğümmleri listeler.
- **roscall ping node_name**: Düğüme olan bağlantıyı test eder.

Parametre Servisi (Parameter Service)

- Parametreler ile, çalışan düğümleri yapılandırmak veya bir düğümün çalışma parametrelerini değiştirmek mümkündür.
- ROS, Parametre Server ile çalışmak için *rosparam* aracına sahiptir. *rosparam* aracında desteklenen bazı komutlar:
 - **rosparam list**: Sunucudaki tüm parametreleri listeler.
 - **rosparam get parameter**: Bir parametrenin değerini alır.
 - **rosparam set parameter**: Bir parametrenin değerini ayarlar.
 - **rosparam delete parameter**: Bir parametreyi siler.
 - **rosparam dump file**: Parametre sunucusunu bir dosyaya kaydeder.
 - **rosparam load file**: Parametre sunucusunda bir dosyayı (parametreleriyle birlikte) yükler.

Mesajlar (Messages) – 1/2

- Düğümmler birbirleriyle mesajlar yoluyla haberleşir. Bir mesaj diğer düğümmlere bilgi sağlayan veri içerir.
- Bir mesaj **tip** ve **isim** olmak üzere iki parçadan oluşmaktadır.
- Kendi mesaj tipimizi oluşturabiliriz.

In ROS, you can find a lot of standard types to use in messages, as shown in the following table list:

Primitive type	Serialization	C++	Python
bool (1)	unsigned 8-bit int	uint8_t (2)	bool
int8	signed 8-bit int	int8_t	int
uint8	unsigned 8-bit int	uint8_t	int (3)
int16	signed 16-bit int	int16_t	int
uint16	unsigned 16-bit int	uint16_t	int
int32	signed 32-bit int	int32_t	int
uint32	unsigned 32-bit int	uint32_t	int
int64	signed 64-bit int	int64_t	long
uint64	unsigned 64-bit int	uint64_t	long
float32	32-bit IEEE float	float	float
float64	64-bit IEEE float	double	float
string	ascii string (4)	std::string	string
time	secs/nsecs signed 32-bit ints	ros::Time	rospy.Time
duration	secs/nsecs signed 32-bit ints	ros::Duration	rospy.Duration

Mesajlar (Messages) – 2/2

Başlıklar (Headers) ROS'ta özel bir tiptir. Zaman çizelgesi, mesajların kimden geldiğini öğrenebilmemizi sağlayan numaralandırma sistemi ve dizi numarasına sahiptir.

[std_msgs/Header Message](#)

File: `std_msgs/Header.msg`

Raw Message Definition

```
# Standard metadata for higher-level stamped data types.
# This is generally used to communicate timestamped data
# in a particular coordinate frame.
#
# sequence ID: consecutively increasing ID
uint32 seq
# Two-integer timestamp that is expressed as:
# * stamp.sec: seconds (stamp_secs) since epoch (in Python the variable is called 'secs')
# * stamp.nsec: nanoseconds since stamp_secs (in Python the variable is called 'nsecs')
# time-handling sugar is provided by the client library
time stamp
#Frame this data is associated with
string frame_id
```

Compact Message Definition

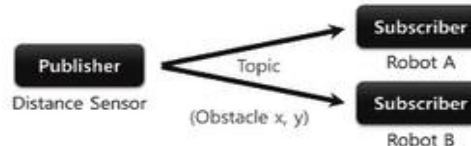
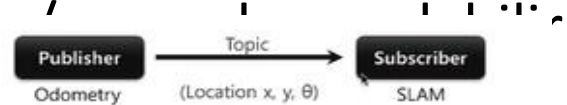
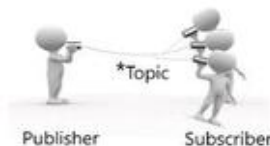
```
uint32 seq
time stamp
string frame_id
```

ROS mesaj tanımını ve mesaj tipinin belirtildiği kaynak dosyasını görebilmemizi sağlayan *rosmmsg* isimli bir araca sahiptir. *rosmmsg* aracında desteklenen bazı komutlar:

- **rosmmsg show:** Bu mesajın alanlarını görüntüler.
- **rosmmsg list:** Tüm mesajları listeler
- **rosmmsg package:** Belirli paketteki tüm mesajları listeler.
- **rosmmsg packages:** Mesajı olan tüm paketleri listeler.
- **rosmmsg users:** Mesaj türünü kullanan kod dosyalarını arar.

Konular (Topics) – 1/2

- Mesajlar, yayınlama / abone olma semantiğine sahip bir taşıma sistemi aracılığıyla yönlendirilir. Bir düğüm, belirli bir konu yayınlayarak bir mesaj gönderir.
- Konu, mesajın içeriğini tanımlamak için kullanılan bir addır.
- Belirli bir veri türüyle ilgilenen bir düğüm uygun konuya abone olacaktır.
- Karışıklıktan kaçınmak için konu adlarının benzersiz olması önemlidir.
- Tek bir konu için birden fazla eşzamanlı yayıncı ve abone olabilir ve tek bir düğüm birden fazla konuya ya



Konular (Topics) – 2/2

ROS, *rostopic* denilen konularda çalışacak bir araca sahiptir. *rostopic* aracında desteklenen bazı komutlar :

- **rostopic bw/topic**: Konunun kullandığı bant genişliğini gösterir.
- **rostopic echo/topic**: Mesajları ekrana basar.
- **rostopic find message_type**: Türlerine göre konuları bulur.
- **rostopic hz/topic**: Konunun yayınlanma oranını gösterir.
- **rostopic info/topic**: Mesaj türü, yayıncıları ve aboneleri gibi konu hakkındaki bilgileri yazdırır.
- **rostopic list**: Aktif konular hakkında bilgi yazdırır.
- **rostopic pub/topic type args**: Konuyla ilgili verileri yayınlar. İstedığımız konuda, doğrudan komut satırından veri oluşturmamızı ve yayınlamamızı sağlar.
- **rostopic type/topic**: Konu türünü, yani yayınladığı mesaj türünü yazdırır.

Servisler (Services)

- Yayınlama / abone olma modeli çok esnek bir iletişim paradigmasıdır, ancak çoktan çoğa, tek yönlü taşımacılığı genellikle dağıtılmış bir sistemde istenen istek / cevap etkileşimleri için uygun değildir.
- Talep / cevap, bir çift mesaj yapısı tarafından tanımlanan servisler vasıtasıyla yapılır: biri istek (request) için, diğeri yanıt (response) için.
- Sağlayıcı bir isim altında bir servis sunar ve bir müşteri talep mesajını gönderip cevabı bekleyerek servisi kullanır.

ROS servislerle çalışmak için *rossrv* ve *rosservice* komut satırı araçlarına sahiptir. Bu araçlarda desteklenen bazı komutlar:

- **rosservice call/service args:** Servisi verilen argümanlarla çağırır.
- **rosservice find msg-type:** Servis tipine göre servis bulur.
- **rosservice info/service:** Servis hakkındaki bilgileri yazdırır.
- **rosservice list:** Aktif servisleri listeler.
- **rosservice type/servis:** Servis tipini yazdırır.

Çantalar (Bags)

- Çanta, ROS tarafından oluşturulan, tüm mesajların, konuların, servislerin ve diğer bilgilerin tüm bilgilerini kaydetmek ve daha sonra oynatmak için **.bag** biçiminde oluşturulan bir dosyadır.
- Çantalar, toplanması zor olabilen ancak algoritmalar geliştirmek ve test etmek için gerekli olan sensör verileri gibi verileri depolamak için önemli bir mekanizmadır.
- Çanta dosyalarını kullanmak için ROS'ta kullanılabilecek araçlar:
 - **rosbag**: İstenilen verileri kaydetmek, oynatmak ve gerçekleştirmek için kullanılır.
 - **rqt_bag**: Verileri grafik ortamda görselleştirmek için kullanılır.

Ana (Master) – 1/2

- ROS'taki düğümlerin birbirleriyle iletişim kurmasını kolaylaştıran kısmına ROS master denir.
- ROS Master, İşlem Grafiğinin geri kalanına arama sağlar. Master olmadan, düğümler birbirlerini bulamaz, mesaj alışverişinde bulunamaz veya servis çağrısı yapamazlar.
- Herhangi bir ROS düğümünü çalıştırmadan önce, ROS Master ve ROS parametre sunucusunu başlatmalıyız. ROS Master ve ROS parametre sunucusunu *roscore* adlı tek bir komut kullanarak başlatabiliriz.

Ana (Master) – 2/2

```
robot@robot-VirtualBox:~$ roscore
... logging to /home/robot/.ros/log/a3a8e160-e1ae-11e4-b7be-0800273c354c/roslaunch-robot-VirtualBox-2138.log
Checking log directory for disk usage. This may take awhile.
Press Ctrl-C to interrupt
Done checking log file disk usage. Usage is <1GB.

started roslaunch server http://robot-VirtualBox:42377/
ros_comm version 1.11.10

SUMMARY
=====

PARAMETERS
* /rostdistro: indigo
* /rosversion: 1.11.10

NODES

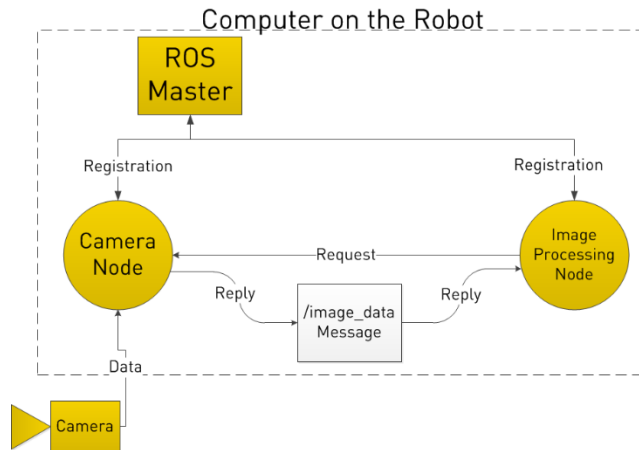
auto-starting new master
process[master]: started with pid [2183]
ROS_MASTER_URI=http://robot-VirtualBox:11311/

setting /run_id to a3a8e160-e1ae-11e4-b7be-0800273c354c
process[rosout-1]: started with pid [2196]
started core service [/rosout]
```

- Birinci bölümde, ROS düğümlerinden gelen günlükleri toplamak için ~/.ros/log klasörünün içinde bir günlük dosyası oluşturulur. Bu dosya hata ayıklama amacıyla kullanılabilir.
- İkinci bölümde, roscore adlı bir ROS başlatma dosyası başlatılır. Bu bölüm port içindeki ROS parametre sunucusunun adresini gösterir.
- Üçüncü bölümde, rostdistro ve rosversion gibi parametreleri görüntülenmektedir.
- Dördüncü bölümde, rosmaster düğümünün daha önce ortam değişkeni olarak tanımladığımız ROS_MASTER_URI kullanılarak başlatıldığı görülmektedir.
- Beşinci bölümde, rosout düğümünün başladığı görülmektedir.

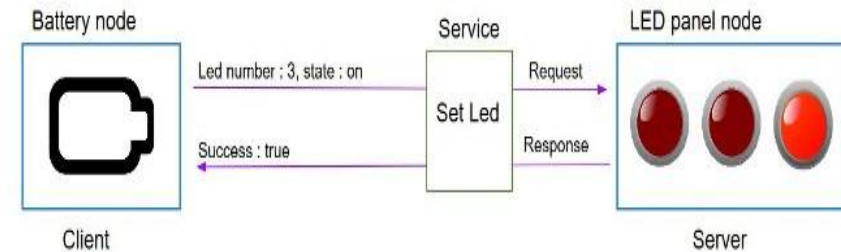
Publish-Subscribe

Bu iletişim modeli, *yayımcının* açıkça alıcıları belirtmeden ya da amaçlanan alıcıların bilgisine sahip olmadan mesajın yayınlanmasını gerektirir. *Abone* yayınlanan mesajlardan ilgili olanları kaydeder.

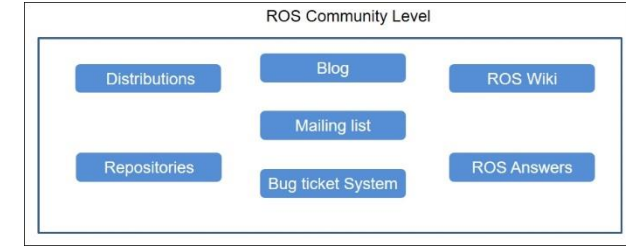


Service-Client

Tek seferlik iletişim sağlayan ve müşterinin istek gönderip, sunucunun geriye yanıt döndürdüğü iletişim modelidir. Robotun özel bir görev yapması istendiğinde (örneğin A noktasından B noktasına yol bulmasında) kullanılır.



Topluluk – Community Level

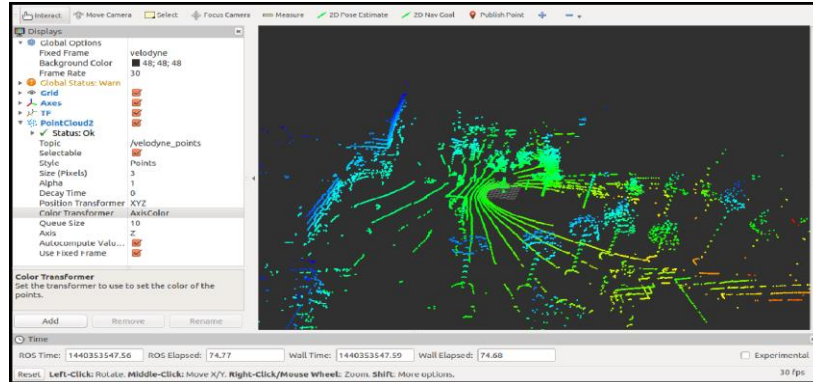


- **Dağıtımlar (Distributions):** ROS Dağıtımları, yükleyebileceğiniz sürümlü yığınların koleksiyonlarıdır.
- **Depolar (Repositories):** ROS, farklı kurumların kendi robot yazılım bileşenlerini geliştirip yayınlayabilecekleri bir kod deposu sunar.
- **ROS Wiki:** ROS hakkındaki bilgileri belgeleyen ana forumdur.
- **Mail Listesi:** Ros-users e-posta listesi, ROS'taki yeni güncellemelerin yanı sıra ROS yazılımı hakkında sorular soran bir forum olan birincil iletişim kanalıdır.
- **ROS Answers:** ROS ile ilgili sorularınızı cevaplamak için soru-cevap sitesidir.
- **Blog:** <http://www.ros.org/news> , fotoğraflar ve videolar dahil olmak üzere düzenli güncellemeler sağlar.

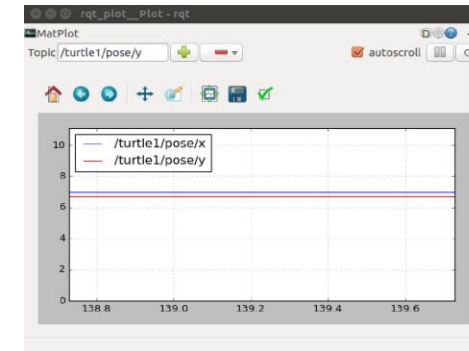
ROS Araçları

ROS, mesajları incelemek ve hata ayıklamak için çeşitli GUI ve komut satırı araçlarına sahiptir. Bunlardan bazıları:

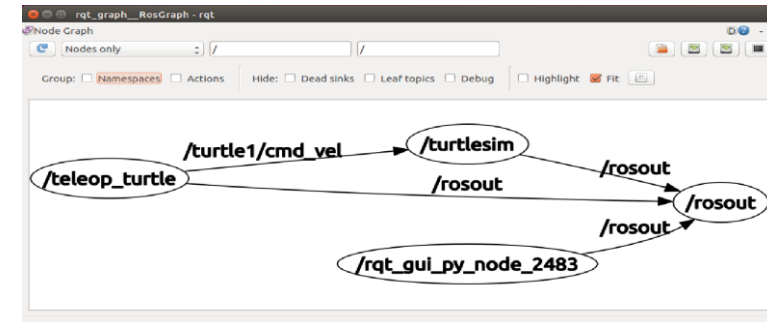
- **rviz:** ROS konularından ve parametrelerinden 2D ve 3D değerlerini görselleştirmek için ROS'ta mevcut 3D görselleştiricilerden biridir.



- **rqt_plot:** ROS konuları biçimindeki skaler değerleri çizmek için bir araçtır.



- **rqt_graph:** ROS düğümleri arasındaki bağlantı grafiğini görselleştirir.



İçerik – 13.00-14.45

- Uygulama 1: ROS Ortamının Hazırlanması
- Uygulama 2: catkin Paketi Oluşturma ve ROS Ortamını Tanıma
- Uygulama 3: TurtleSim
- Uygulama 4: Mesaj ve Servis Oluşturma
- Uygulama 5: Publisher-Subscriber Uygulaması
- Uygulama 6: Service-Client Uygulaması
- Uygulama 7: Verileri Kaydetme ve Oynatma

Uygulama 1: ROS Ortamının Hazırlanması – 1/3

Çalışma alanı paketleri içeren bir klasördür. Bu paketler kaynak dosyaları içerir. Çeşitli paketler aynı anda derlenmek (merkeziştirmek) istendiğinde kullanışlıdır.

```
çalışma_alanı_klasörü/ -- ÇALIŞMA ALANI
src/                  -- KAYNAK ALANI
CMakeLists.txt       -- catkin'in oluşturduğu 'ana' CMake dosyası
paket_1/
  CMakeLists.txt      -- paket_1 için CMakeLists.txt dosyası
  package.xml         -- paket_1 için package.xml dosyası
...
paket_n/
  CMakeLists.txt      -- paket_n için CMakeLists.txt dosyası
  package.xml         -- paket_n için package.xml dosyası
```

```
catkin_ws
├── build
│   ├── catkin
│   ├── catkin_generated
│   └── Makefile
├── devel
│   ├── setup.zsh
│   └── ...
└── src
    ├── CMakeLists.txt -> /opt/ros/kinetic/share/catkin/cmake/toplevel.cmake
    └── ...
```

- **Kaynak alan (src):** Kaynak alana (src klasörü), paketler, projeler, vb. konulur. Bu alanda ayrıca *CMakeLists.txt* dosyası bulunur.
- **Derleme alanı (build):** derleme klasöründe cmake ve catkin, paketler ve projeler için önbellek bilgilerinin, yapılandırmasını ve diğer ara dosyaları saklar.
- **Geliştirme alanı (devel):** Derlenmiş programları korumak ve programları kurulum aşaması olmadan test etmek için kullanılır.

Uygulama 1: ROS Ortamının Hazırlanması – 2/3

1. Ortamı kontrol edelim:

printenv | grep ROS

```
moguztas@moguztas: ~  
moguztas@moguztas: ~ 80x24  
moguztas@moguztas:~$ printenv | grep ROS  
ROS_ROOT=/opt/ros/kinetic/share/ros  
ROS_PACKAGE_PATH=/home/moguztas/hd_map_ws/src:/opt/ros/kinetic/share  
ROS_MASTER_URI=http://localhost:11311  
ROSLISP_PACKAGE_DIRECTORIES=/home/moguztas/hd_map_ws/devel/share/common-lisp  
ROS_DISTRO=kinetic  
ROS_ETC_DIR=/opt/ros/kinetic/etc/ros  
moguztas@moguztas:~$
```

2. Konfigürasyon ayarlarımızı her seferinde yapmamak için:

gedit ~/.bashrc

Gedit editöründe açılan ekrana aşağıdaki kodlar eklenir.

<source /opt/ros/kinetic/setup.bash>

<source /home/<kullanıcı_adı>/ros_ws/devel/setup.bash>

Not: bashrc'de yapılan değişikliklerin önceden açılan terminalde algılanabilmesi için *bash* komutu ile terminal yenilenmelidir.

```
.bashrc (~) - gedit  
Open  [?]  CMakeLists.txt  .bashrc  
case $TERM in  
xterm*) rxvt*)  
PS1="\[\e]0;$[debian_chroot:+($debian_chroot)]\u@h: \w[a]SP$1"  
;;  
*)  
;;  
esac  
  
# enable color support of ls and also add handy aliases  
if [ -x /usr/bin/dircolors ]; then  
test -r ~/.dircolors && eval "$(dircolors -b ~/.dircolors)" || eval "$(dircolors -b)"  
alias ls='ls --color=auto'  
alias dir='dir --color=auto'  
alias vdir='vdir --color=auto'  
  
alias grep='grep --color=auto'  
alias fgrep='fgrep --color=auto'  
alias egrep='egrep --color=auto'  
fi  
  
# colored GCC warnings and errors  
#export GCC_COLORS='error=01;31:warning=01;35:note=01;36:caret=01;32:locus=01:quote=01'  
  
# some more ls aliases  
alias ll='ls -lF'  
alias la='ls -A'  
alias l='ls -CF'  
  
# Add an "alert" alias for long running commands. Use like so:  
# sleep 10; alert  
alias alert='notify-send --urgency=low -i "${@} $?" && echo terminal || echo error' "${history|tail -n1|sed -e '\s/[0-9]\{1,\}/s//;/s/[;]&]\s*alert$/\s*'\''"  
  
# Alias definitions.  
# You may want to put all your additions into a separate file like  
# ~/.bash_aliases, instead of adding them here directly.  
# See /usr/share/doc/bash-doc/examples in the bash-doc package.  
  
if [ -f ~/.bash_aliases ]; then  
. ~/.bash_aliases  
fi  
  
# enable programmable completion features (you don't need to enable  
# this, if it's already enabled in /etc/bash.bashrc and /etc/profile  
# sources /etc/bash.bashrc).  
if ! shopt -oq posix; then  
if [ -f /usr/share/bash-completion/bash_completion ]; then  
. /usr/share/bash-completion/bash_completion  
elif [ -f /etc/bash_completion ]; then  
. /etc/bash_completion  
fi  
fi  
  
source /opt/ros/kinetic/setup.bash  
source /home/moguztas/ros_ws/devel/setup.bash  
export ROS_HOSTNAME=moguztas  
export ROS_MASTER_URI=http://localhost:11311  
export ROS_PACKAGE_PATH=/home/moguztas/ros_ws/src
```

Uygulama 1: ROS Ortamının Hazırlanması – 3/3

3. Çalışma ortamı oluşturmak için:

```
mkdir -p ~/ros_ws/src
cd ~/ros_ws/
catkin_make
```

Not: catkin_make ile aynı işe yapabilecek kod bloğu:

```
cd ~/ros_ws
cd src
catkin_init_workspace
cd ..
mkdir build
cd build
cmake ../src -DCMAKE_INSTALL_PREFIX=../install -DCATKIN_DEVEL_PREFIX=../devel
make
```

Not: ROS_PACKAGE_PATH yapılandırma değişkeninin konumu için terminal ekranına aşağıdaki kod yazılır.

```
echo $ROS_PACKAGE_PATH
/home/(KULLANICI_ADINIZ)/ros_ws/src:/opt/ros/kinetic/share
```

```
moguztas@moguztas:~$ mkdir -p ~/ros_ws/src
moguztas@moguztas:~$ cd ros_ws/
moguztas@moguztas:~/ros_ws$ catkin_make
Base path: /home/moguztas/ros_ws
Source space: /home/moguztas/ros_ws/src
Build space: /home/moguztas/ros_ws/build
Devel space: /home/moguztas/ros_ws/devel
Install space: /home/moguztas/ros_ws/install
Creating symlink "/home/moguztas/ros_ws/src/CMakeLists.txt" pointing to "/opt/ros/kinetic/share/catkin/cmake/toplevel.cmake"
####
#### Running command: "cmake /home/moguztas/ros_ws/src -DCATKIN_DEVEL_PREFIX=/home/moguztas/ros_ws/devel -DCMAKE_INSTALL_PREFIX=/home/moguztas/ros_ws/install -G Unix Makefiles" in "/home/moguztas/ros_ws/build"
####
-- The C compiler identification is GNU 5.4.0
-- The CXX compiler identification is GNU 5.4.0
-- Check for working C compiler: /usr/bin/cc
-- Check for working C compiler: /usr/bin/cc -- works
-- Detecting C compiler ABI info
-- Detecting C compiler ABI info - done
-- Detecting C compile features
-- Detecting C compile features - done
-- Check for working CXX compiler: /usr/bin/c++
-- Check for working CXX compiler: /usr/bin/c++ -- works
-- Detecting CXX compiler ABI info
-- Detecting CXX compiler ABI info - done
-- Detecting CXX compile features
-- Detecting CXX compile features - done
-- Using CATKIN_DEVEL_PREFIX: /home/moguztas/ros_ws/devel
-- Using CMAKE_PREFIX_PATH: /home/moguztas/hd_map_ws/devel;/opt/ros/kinetic
-- This workspace overlays: /home/moguztas/hd_map_ws/devel;/opt/ros/kinetic
-- Found PythonInterp: /usr/bin/python (found version "2.7.12")
-- Using PYTHON_EXECUTABLE: /usr/bin/python
-- Using Debian Python package layout
-- Using empy: /usr/bin/empy
-- Using CATKIN_ENABLE_TESTING: ON
-- Call enable_testing()
-- Using CATKIN_TEST_RESULTS_DIR: /home/moguztas/ros_ws/build/test_results
-- Looking for pthread.h
-- Looking for pthread.h - found
-- Looking for pthread_create
-- Looking for pthread_create - not found
-- Looking for pthread_create in pthreads
-- Looking for pthread_create in pthreads - not found
-- Looking for pthread_create in pthread
-- Looking for pthread_create in pthread - found
-- Found Threads: TRUE
-- Found gtest sources under '/usr/src/gtest': gtests will be built
-- Using Python nosetests: /usr/bin/nosetests-2.7
-- catkin 0.7.8
-- BUILD_SHARED_LIBS is on
-- Configuring done
-- Generating done
-- Build files have been written to: /home/moguztas/ros_ws/build
####
#### Running command: "make -j8 -l8" in "/home/moguztas/ros_ws/build"
####
moguztas@moguztas:~/ros_ws$
```


Uygulama 2: catkin Paketi Oluşturma ve ROS Ortamını Tanıma – 1/5

1. İlk olarak, ros çalışma alanında src dizinine gidilir.

`cd ~/ros_ws/src`

2. `catkin_create_pkg` komutu ile `beginner_tutorials` isimli `std_msgs`, `roscpp` ve `rospy` a bağlı olan bir paket oluşturulur:

`catkin_create_pkg beginner_tutorials std_msgs rospy roscpp`

Not: Bu beginner_tutorial isimli `package.xml` ve bir `CMakeLists.txt` dosyası içeren bir dosya oluşturacaktır. `CMakeLists.txt` dosyası kısmen `catkin_create_pkg` komutu tarafından doldurulmuştur.

```
moguztas@moguztas:~/ros_ws/src$ catkin_create_pkg beginner_tutorials std_msgs rospy roscpp
Created file beginner_tutorials/package.xml
Created file beginner_tutorials/CMakeLists.txt
Created folder beginner_tutorials/include/beginner_tutorials
Created folder beginner_tutorials/src
Successfully created files in /home/moguztas/ros_ws/src/beginner_tutorials. Please adjust the values in package.xml.
```

3. beginner tutorials klasörü içerisinde ne olduğuna bakmak için:

`cd beginner_tutorials`

`ls`

```
moguztas@moguztas:~/ros_ws/src$ cd beginner_tutorials/
moguztas@moguztas:~/ros_ws/src/beginner_tutorials$ ls
CMakeLists.txt  include  package.xml  src
```

Uygulama 2: catkin Paketi Oluşturma ve ROS Ortamını Tanıma – 2/5

4. `beginner_tutorials` src dosyasına giderek burada basit bir kod yazalım:

```
cd src
```

```
gedit first_script.cpp
```

5. CMakeLists.txt dosyasında yazdığımız kodun çalıştırılabilir olmasını sağlayalım:

cd ..

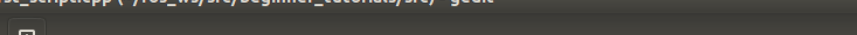
```
gedit CMakeLists.txt
```

6. Çalışma alanımızı derleyelim.

```
cd ~/ros_ws/
```

catkin make

Not: Derleme tamamlandığında src klasöründe build,devel ve src alt klasörleri kurulmuş olacak,paket kullanıma hazır hale gelecektir.



```
first_script.cpp (~/.ros_ws/src/beginner_tutorials/src) - gedit
Open Save
#include <iostream>
using namespace std;
int main()
{
    cout << "ROS Uygulamali Egitim" << endl;
    return 0;
}
```

```
CMakeLists.txt (~/ros_ws/src/beginner_tutorials - gedit)
Open Save
# LIBRARIES beginner_tutorials
# CATKIN_DEPENDS roscpp rospy std_msgs
# DEPENDS system_lib
)

#####
## Build ##
#####

## Specify additional locations of header files
## Your package locations should be listed before other locations
include_directories(
  # include
  ${catkin_INCLUDE_DIRS}
)

## Declare a C++ library
# add_library(${PROJECT_NAME})
# set_target_properties(${PROJECT_NAME} PROPERTIES OUTPUT_NAME node PREFIX "")

## Add cmake target dependencies of the library
## as an example, code may need to be generated before libraries
## either from message generation or dynamic reconfigure
# add_dependencies(${PROJECT_NAME} ${${PROJECT_NAME}_EXPORTED_TARGETS} ${catkin_EXPORTED_TARGETS})

## Declare a C++ executable
## With catkin make all packages are built within a single CMake context
## The recommended prefix ensures that target names across packages don't collide
# add_executable(${PROJECT_NAME}_node src/beginner_tutorials_node.cpp)
# set_target_properties(${PROJECT_NAME}_node PROPERTIES OUTPUT_NAME node PREFIX "")

## Add cmake target dependencies of the executable
## same as for the library above
# add_dependencies(${PROJECT_NAME}_node ${${PROJECT_NAME}_EXPORTED_TARGETS} ${catkin_EXPORTED_TARGETS})

## Specify libraries to link a library or executable target against
target_link_libraries(beginner_tutorials_node
  ${catkin_LIBRARIES}
)

moguztas@moguztas:~/ros_ws$ catkin_make
Base path: /home/moguztas/ros_ws
Source space: /home/moguztas/ros_ws/src
Build space: /home/moguztas/ros_ws/build
Devel space: /home/moguztas/ros_ws/devel
Install space: /home/moguztas/ros_ws/install
####
### Running command: "make cmake_check_build_system" in "/home/moguztas/ros_ws/build"
###
-- Using CATKIN_DEVEL_PREFIX: /home/moguztas/ros_ws/devel
-- Using CMAKE_PREFIX_PATH: /home/moguztas/hd_map_ws/devel;/opt/ros/kinetic
-- This workspace overlays: /home/moguztas/hd_map_ws/devel;/opt/ros/kinetic
-- Using PYTHON_EXECUTABLE: /usr/bin/python
-- Using Debian Python package layout
-- Using empy: /usr/bin/empy
-- Using CATKIN_ENABLE_TESTING: ON
-- Call enable_testing()
-- Using CATKIN_TEST_RESULTS_DIR: /home/moguztas/ros_ws/build/test_results
-- Found gtest sources under '/usr/src/gtest': gtests will be built
-- Using Python nosetests: /usr/bin/nosetests-2.7
-- catkin 0.7.8
-- BUILD_SHARED_LIBS is on
-----
-- traversing 1 packages in topological order:
-- - beginner_tutorials
-----
+++ processing catkin package: 'beginner_tutorials'
==> add_subdirectory(beginner_tutorials)
-- Configuring done
-- Generating done
-- Build files have been written to: /home/moguztas/ros_ws/build
####
### Running command: "make -j8 -l8" in "/home/moguztas/ros_ws/build"
###
Scanning dependencies of target beginner_tutorials_node
[ 50%] Building CXX object beginner_tutorials/CMakeFiles/beginner_tutorials_node.dir/src/first_script.cpp.o
[100%] Linking CXX executable /home/moguztas/ros_ws/devel/lib/beginner_tutorials/beginner_tutorials_node
[100%] Built target beginner_tutorials_node
moguztas@moguztas:~/ros_ws$
```

Uygulama 2: catkin Paketi Oluşturma ve ROS Ortamını Tanıma – 3/5

7. Yazılan kodun çalıştırılması için iki terminal açılır. İlk terminalde aşağıdaki kod çalıştırılır.

roscore

```
moguztas@moguztas:~$ roscore
... logging to /home/moguztas/.ros/log/50b06bfc-cb71-11e9-b368-60f6774b2981/roslaunch-moguztas-8219.log
Checking log directory for disk usage. This may take awhile.
Press Ctrl-C to interrupt
Done checking log file disk usage. Usage is <1GB.

started roslaunch server http://moguztas:35969/
ros_comm version 1.12.12

SUMMARY
=====

PARAMETERS
* /rostdistro: kinetic
* /rosversion: 1.12.12

NODES

auto-starting new master
process[master]: started with pid [8235]
ROS_MASTER_URI=http://moguztas:11311/

setting /run_id to 50b06bfc-cb71-11e9-b368-60f6774b2981
process[rosout-1]: started with pid [8248]
started core service [/rosout]
```

8. Diğer terminalde ise oluşturulan ros paketi çalıştırılır.

roslaunch beginner_tutorials beginner_tutorials_node

```
moguztas@moguztas:~$ roslaunch beginner_tutorials beginner_tutorials_node
ROS Uygulamalı Eğitim
moguztas@moguztas:~$
```

Uygulama 2: catkin Paketi Oluşturma ve ROS Ortamını Tanıma – 4/5

9.1. *rospack* aracı ile paketteki bağımlılıkları görüntülemek için:

rospack depends1 beginner_tutorials

```
moguztas@moguztas:~/ros_ws$ rospack depends1 beginner_tutorials
roscpp
rospy
std_msgs
moguztas@moguztas:~/ros_ws$
```

9.2. Bu bağımlılıklar *package.xml* dosyasında da listelenmektedir.

roscd beginner_tutorials

gedit package.xml

```
<buildtool_depend>catkin</buildtool_depend>
<build_depend>roscpp</build_depend>
<build_depend>rospy</build_depend>
<build_depend>std_msgs</build_depend>
<build_export_depend>roscpp</build_export_depend>
<build_export_depend>rospy</build_export_depend>
<build_export_depend>std_msgs</build_export_depend>
<exec_depend>roscpp</exec_depend>
<exec_depend>rospy</exec_depend>
<exec_depend>std_msgs</exec_depend>

<!-- The export tag contains other, unspecified, tags -->
<export>
  <!-- Other tools can request additional information be placed here -->

</export>
</package>
```

Uygulama 2: catkin Paketi Oluşturma ve ROS Ortamını Tanıma – 5/5

10. Dolaylı bağımlılıklar *rospack* aracıyla görüntülenebilir. Örneğin *rospy* a bağlı bağımlılıkları görmek için:

rospack depends1 rospy

```
moguztas@moguztas:~/ros_ws/src/beginner_tutorials$ rospack depends1 rospy
genpy
roscpp
rosgraph
rosgraph_msgs
roslib
std_msgs
moguztas@moguztas:~/ros_ws/src/beginner_tutorials$
```

11. Paketteki tüm bağımlılıkları görmek için:

rospack depends beginner_tutorials

```
moguztas@moguztas:~/ros_ws/src/beginner_tutorials$ rospack depends beginner_tutorials
cpp_common
rostime
roscpp_traits
roscpp_serialization
catkin
genmsg
genpy
message_runtime
gencpp
geneus
genodejs
genlisp
message_generation
roscpp
rosgraph
rospack
roslib
rospy
moguztas@moguztas:~/ros_ws/src/beginner_tutorials$
```

Uygulama 3: TurtleSim – 1/4

1. TurtleSim uygulaması için:

sudo apt-get install ros-kinetic-ros-tutorials

2. ROS Master'ı başlatmak için:

roscore

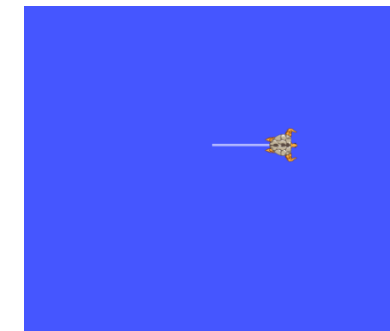
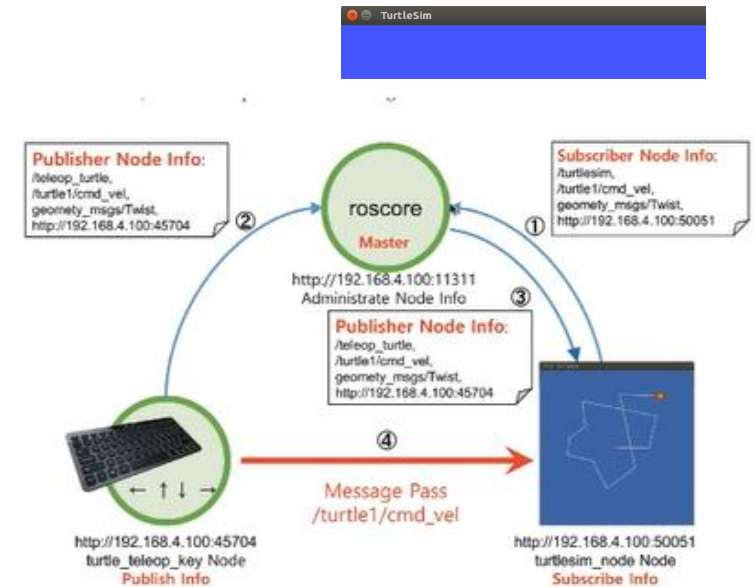
3. TurtleSim'i çalıştırmak için:

roslaunch turtlesim turtlesim_node

4. Klavye ile kontrol için:

roslaunch turtlesim turtle_teleop_key

```
moguztas@moguztas:~$ roslaunch turtlesim turtlesim_node
Gtk-Message: Failed to load module "overlay-scrollbar"
[ INFO] [1567202501.052899863]: Starting turtlesim with node name /turtlesim
[ INFO] [1567202501.071707138]: Spawning turtle [turtle1] at x=[5.544445], y=[5.544445], theta=[0.000000]
```



Uygulama 3: TurtleSim – 2/4

5. *roscat list*

```
moguztas@moguztas:~$ roscat list
/rosout
/teleop_turtle
/turtlesim
moguztas@moguztas:~$
```

6. *rostopic list*

```
moguztas@moguztas:~$ rostopic list
/rosout
/rosout_agg
/turtle1/cmd_vel
/turtle1/color_sensor
/turtle1/pose
moguztas@moguztas:~$
```

7. *rostopic info /turtle1/cmd_vel*

```
moguztas@moguztas:~$ rostopic info /turtle1/cmd_vel
Type: geometry_msgs/Twist

Publishers:
 * /teleop_turtle (http://moguztas:32976/)

Subscribers:
 * /turtlesim (http://moguztas:44493/)

moguztas@moguztas:~$
```

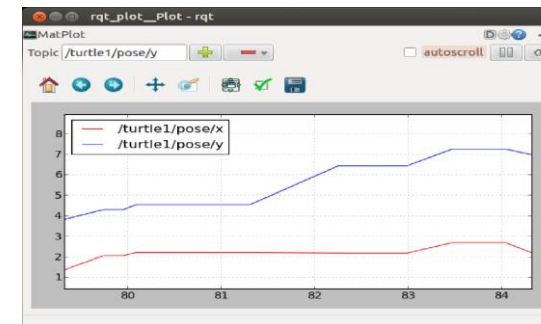
8. *rosmmsg show geometry_msgs/Twist*

```
moguztas@moguztas:~$ rosmmsg show geometry_msgs/Twist
geometry_msgs/Vector3 linear
float64 x
float64 y
float64 z
geometry_msgs/Vector3 angular
float64 x
float64 y
float64 z
moguztas@moguztas:~$
```

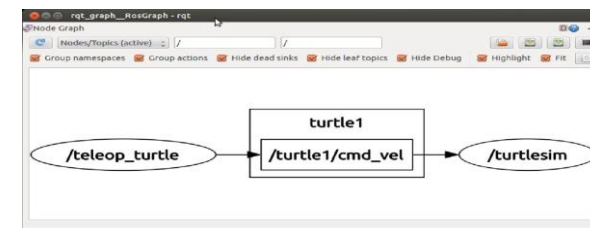
9. *rostopic echo /turtle1/cmd_vel*

```
moguztas@moguztas:~$ rostopic echo /turtle1/cmd_vel
linear:
  x: 2.0
  y: 0.0
  z: 0.0
angular:
  x: 0.0
  y: 0.0
  z: 0.0
---
```

10. *rqt_plot*



11. *rqt_graph*



Uygulama 3: TurtleSim – 3/4

12. *rosservice list*

```
moguztas@moguztas:~$ 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
moguztas@moguztas:~$
```

13. *rosservice type/spawn*

```
moguztas@moguztas:~$ rosservice type /spawn
turtlesim/Spawn
moguztas@moguztas:~$
```

14. *rossrv show turtlesim/Spawn*

```
moguztas@moguztas:~$ rossrv show turtlesim/Spawn
float32 x
float32 y
float32 theta
string name
---
string name
moguztas@moguztas:~$
```

15. *rosservice call /spawn 3 3 0 new_turtle*

```
moguztas@moguztas:~$ rosservice call /spawn 3 3 0 new_turtle
name: "new_turtle"
moguztas@moguztas:~$
```

16. *rosparam list*

```
moguztas@moguztas:~$ rosparam list
/background_b
/background_g
/background_r
/rosdistro
/roslaunch/uris/host_moguztas__35969
/rosversion
/run_id
moguztas@moguztas:~$
```

17. *rosparam get /background_b*

```
moguztas@moguztas:~$ rosparam get /background_b
255
moguztas@moguztas:~$
```

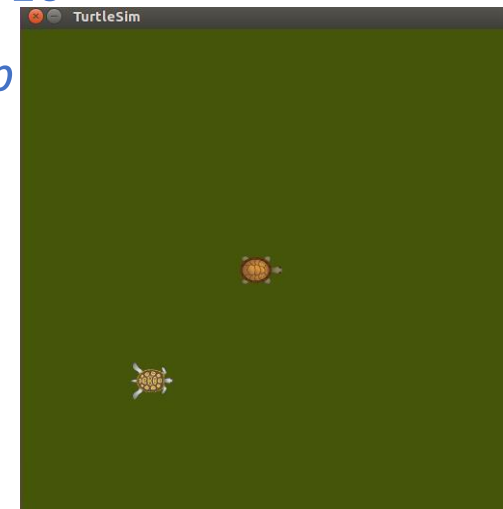
18. *rosparam set /background_b 10*

rosparam get /background_b

```
moguztas@moguztas:~$ rosparam set /background_b 10
moguztas@moguztas:~$ rosparam get /background_b
10
moguztas@moguztas:~$
```

19. *rosservice call /clear*

```
moguztas@moguztas:~$ rosservice call /clear
moguztas@moguztas:~$
```



Uygulama 3: TurtleSim – 4/4

roslaunch, belirlenmiş çalıştırma dosyasını başlatır. Kullanımı aşağıdaki gibidir:

roslaunch [package] [filename.launch]

20. Öncelikle *beginner_tutorials* ismiyle oluşturduğumuz pakete gidelim ve launch klasörü oluşturalım.

roscd beginner_tutorials

mkdir launch

21. *turtle.launch* isimli bir başlatma dosyası oluşturalım.

```
File Edit View Search Tools Documents Help
Open [icon]
<?xml version="1.0"?>
<launch>

  <!-- TurtleSim Döğümü - Kaplumbağanın çağırılması için -->
  <node pkg="turtle_sim" name="sim" type="turtle_sim node"/>

  <!-- Turtle Draw Square Döğümü - Kaplumbağanın sürekli kare çizmesi için -->
  <node pkg="turtle_sim" name="ds" type="draw_square"/>

</launch>
```

22. Launch dosyasını çağırmak için terminale aşağıdaki kodu yazalım.

roslaunch beginner_tutorials turtle.launch

```
moguztas@moguztas:~$ roslaunch beginner_tutorials turtle.launch
... logging to /home/moguztas/.ros/log/5fba2550-cb78-11e9-b368-60f6774b2981/roslaunch-moguztas-12512.log
Checking log directory for disk usage. This may take awhile.
Press Ctrl-C to interrupt
Done checking log file disk usage. Usage is <1GB.

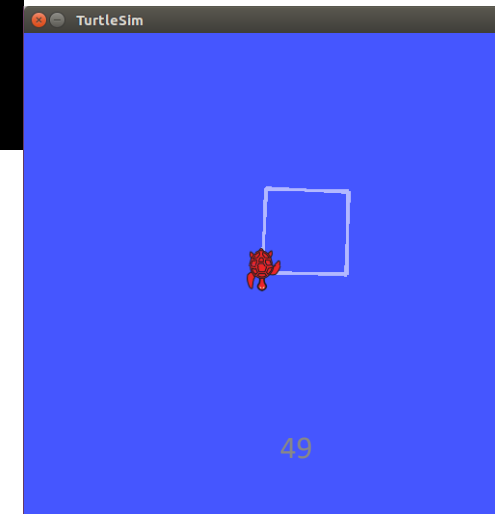
started roslaunch server http://moguztas:37154/

SUMMARY
-----
PARAMETERS
 * /rostdistro: kinetic
 * /rosversion: 1.12.12

NODES
 /
  ds (turtle_sim/draw_square)
  sim (turtle_sim/turtle_sim node)

auto-starting new master
process[master]: started with pid [12522]
ROS_MASTER_URI=http://localhost:11311

setting /run_id to 5fba2550-cb78-11e9-b368-60f6774b2981
process[rosout-1]: started with pid [12535]
started core service [/rosout]
process[sim-2]: started with pid [12542]
process[ds-3]: started with pid [12553]
Gtk-Message: Failed to load module "overlay-scrollbar"
```



Uygulama 4: Mesaj ve Servis Oluşturma – 1/4

Mesaj Oluşturma – 1/2

1. Çalışma alanındaki *beginner_tutorials* klasörüne **msg** klasörü oluşturalım.

```
roscd beginner_tutorials
```

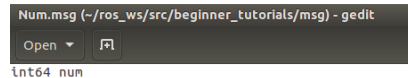
```
mkdir msg
```

2. Mesaj dosyamızı oluşturalım ve terminalde oluşturduğumuz dosyayı gösterelim.

```
echo "int64 num" > msg/Num.msg
```

```
rosmmsg show beginner_tutorials/Num
```

```
moguztas@moguztas:~/ros_ws$ rosmmsg show beginner_tutorials/Num
int64 num
moguztas@moguztas:~/ros_ws$
```



```
Num.msg (~/.ros_ws/src/beginner_tutorials/msg) - gedit
int64 num
```

3. **package.xml** dosyamızı düzenleyelim.

```
roscd beginner_tutorials
```

```
gedit package.xml
```

```
<build_depend>message_generation</build_depend>
```

```
<exec_depend>message_runtime</exec_depend>
```



```
package.xml (~/.ros_ws/src/beginner_tutorials) - gedit
<!-- One license tag required, multiple allowed, one license per tag -->
<!-- Commonly used license strings: -->
<!-- BSD, MIT, Boost Software License, GPLv2, GPLv3, LGPLv2.1, LGPLv3 -->
<license>TODO</license>

<!-- Url tags are optional, but multiple are allowed, one per tag -->
<!-- Optional attribute type can be: website, bugtracker, or repository -->
<!-- Example: -->
<!-- <url type="website">http://wiki.ros.org/beginner_tutorials</url> -->

<!-- Author tags are optional, multiple are allowed, one per tag -->
<!-- Authors do not have to be maintainers, but could be -->
<!-- Example: -->
<!-- <author email="jane.doe@example.com">Jane Doe</author> -->

<!-- The *depend tags are used to specify dependencies -->
<!-- Dependencies can be catkin packages or system dependencies -->
<!-- Examples: -->
<!-- Use depend as a shortcut for packages that are both build and exec dependencies -->
<!-- Note that this is equivalent to the following: -->
<!-- <build_depend>roscpp</build_depend> -->
<!-- <exec_depend>roscpp</exec_depend> -->
<!-- Use build_export_depend for packages you need at compile time: -->
<!-- <build_export_depend>roscpp</build_export_depend> -->
<!-- Use buildtool_depend for build tool packages: -->
<!-- <buildtool_depend>catkin</buildtool_depend> -->
<!-- Use exec_depend for packages you need at runtime: -->
<!-- Use test_depend for packages you need only for testing: -->
<!-- Use doc_depend for packages you need only for building documentation: -->
<!-- <doc_depend>doxygen</doc_depend> -->
<buildtool_depend>catkin</buildtool_depend>
<build_depend>roscpp</build_depend>
<build_export_depend>roscpp</build_export_depend>
<build_export_depend>std_msgs</build_export_depend>
<exec_depend>roscpp</exec_depend>
<exec_depend>std_msgs</exec_depend>
<exec_depend>message_generation</exec_depend>
<exec_depend>message_runtime</exec_depend>

<!-- The export tag contains other, unspecified, tags -->
<export>
```

Uygulama 4: Mesaj ve Servis Oluşturma – 2/4

Mesaj Oluşturma – 2/2

4. CMakeLists.txt dosyamızı aşağıdaki şekilde düzenleyelim.

gedit CMakeLists.txt

```
find_package(catkin REQUIRED COMPONENTS
...
message_generation
)

catkin_package(
...
CATKIN_DEPENDS message_runtime...
...
)

add_message_files(
FILES
Num.msg
)

generate_messages(
DEPENDENCIES
std_msgs
)
```

5. Çalışma alanını derleyin.

*cd ~/ros_ws
catkin_make*

```
~/ros_ws$ catkin_make
Base path: /home/moguztas/ros_ws
Source space: /home/moguztas/ros_ws/src
Build space: /home/moguztas/ros_ws/build
Devel space: /home/moguztas/ros_ws/devel
Install space: /home/moguztas/ros_ws/install
***
*** Running command: "make cmake check build system" in "/home/moguztas/ros_ws/build"
***
-- Using CATKIN_DEVEL_PREFIX: /home/moguztas/ros_ws/devel
-- Using CMAKE_PREFIX_PATH: /home/moguztas/ros_ws/devel;/home/moguztas/rd_map_ws/devel;/opt/ros/kinetic
-- This workspace overlays: /home/moguztas/ros_ws/devel;/home/moguztas/rd_map_ws/devel;/opt/ros/kinetic
-- Using PYTHON_EXECUTABLE: /usr/bin/python
-- Using Debian Python package layout
-- Using empy: /usr/bin/empy
-- Using CATKIN_ENABLE_TESTING: ON
-- Call enable_testing()
-- Using CATKIN_TEST_RESULTS_DIR: /home/moguztas/ros_ws/build/test_results
-- found gtest sources under: /usr/src/gtest: gtests will be built
-- Using Python nosetests: /usr/bin/nosetests-2.7
-- catkin 0.7.8
-- BUILD_SHARED_LIBS is on
--
-- traversing 1 packages in topological order:
-- - beginner_tutorials
--
*** processing catkin package: 'beginner_tutorials'
-- add subdirectory(beginner_tutorials)
-- Using these message generators: gencpp;genmsg;genlisp;genndbjs;genpy
-- beginner_tutorials: 1 messages, 8 services
-- Configuring done
-- Generating done
-- Build files have been written to: /home/moguztas/ros_ws/build
***
*** Running command: "make -j8 -l8" in "/home/moguztas/ros_ws/build"
***
Scanning dependencies of target std_msgs_generate_messages_nodejs
Scanning dependencies of target _beginner_tutorials_generate_messages_check_deps_Num
Scanning dependencies of target std_msgs_generate_messages_ros
Scanning dependencies of target std_msgs_generate_messages_py
Scanning dependencies of target std_msgs_generate_messages_lisp
Scanning dependencies of target beginner_tutorials_node
Scanning dependencies of target std_msgs_generate_messages_cpp
[ 0%] Built target std_msgs_generate_messages_ros
[ 8%] Built target std_msgs_generate_messages_py
[ 9%] Built target std_msgs_generate_messages_lisp
[ 9%] Built target std_msgs_generate_messages_nodejs
[ 14%] Building CXX object beginner_tutorials/CMakeFiles/beginner_tutorials_nodebuild/src/first_script.cpp.o
[ 14%] Built target std_msgs_generate_messages_cpp
[ 14%] Built target _beginner_tutorials_generate_messages_check_deps_Num
Scanning dependencies of target beginner_tutorials_generate_messages_py
Scanning dependencies of target beginner_tutorials_generate_messages_cpp
Scanning dependencies of target beginner_tutorials_generate_messages_nodejs
Scanning dependencies of target beginner_tutorials_generate_messages_lisp
[ 22%] Generating Python from MSG: beginner_tutorials/Num.msg
[ 60%] Generating Java code from beginner_tutorials/Num.msg
[ 60%] Generating C++ code from beginner_tutorials/Num.msg
[ 60%] Generating Lisp code from beginner_tutorials/Num.msg
[ 74%] Built target beginner_tutorials_generate_messages_nodejs
[ 74%] Built target beginner_tutorials_generate_messages_lisp
[ 85%] Linking CXX executable /home/moguztas/ros_ws/devel/lib/beginner_tutorials/beginner_tutorials_node
[100%] Generating Python msg _init_.py for beginner_tutorials
[100%] Built target beginner_tutorials_generate_messages_cpp
[100%] Built target beginner_tutorials_node
[100%] Built target beginner_tutorials_generate_messages_py
[100%] Built target beginner_tutorials_generate_messages_ros
```

Uygulama 4: Mesaj ve Servis Oluşturma – 3/4

Servis Oluşturma – 1/2

1. Çalışma alanındaki *beginner_tutorials* klasörüne **srv** klasörü oluşturalım.

```
roscd beginner_tutorials
```

```
mkdir srv
```

2. srv dosyamızı oluşturalım ve terminalede oluşturduğumuz dosyayı gösterelim.

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

```
rossrv show beginner_tutorials/AddTwoInts
```

```
moguztas@moguztas:~/ros_ws$ rossrv show beginner_tutorials/AddTwoInts
int64 a
int64 b
---
int64 sum
moguztas@moguztas:~/ros_ws$
```



Not: **srv** dosyaları da iki bölüm içermesi hariç **msg** dosyaları gibidirler.

3. package.xml dosyamızı düzenleyelim.

```
roscd beginner_tutorials
```

```
gedit package.xml
```

```
<build_depend>message_generation</build_depend>
```

```
<exec_depend>message_runtime</exec_depend>
```

Uygulama 4: Mesaj ve Servis Oluşturma – 4/4

Servis Oluşturma – 2/2

4. CMakeLists.txt dosyamızı aşağıdaki şekilde düzenleyelim.

```
gedit CMakeLists.txt
```

```
find_package(catkin REQUIRED COMPONENTS
```

```
...  
message_generation  
)
```

```
catkin_package(  
...  
CATKIN_DEPENDS message_runtime ...  
...  
)
```

```
add_service_files(  
FILES  
AddTwoInts.srv  
)
```

```
generate_messages(  
DEPENDENCIES  
std_msgs  
)
```

5. Çalışma alanını derleyin.

```
cd ~/ros_ws  
catkin_make
```

```
moquitas@moquitas:~/ros_ws$ catkin make  
Base path: /home/moquitas/ros_ws  
Source space: /home/moquitas/ros_ws/src  
Build space: /home/moquitas/ros_ws/build  
Devel space: /home/moquitas/ros_ws/devel  
Install space: /home/moquitas/ros_ws/install  
####  
Running command: "make cmake_check_build_system" in "/home/moquitas/ros_ws/build"  
####  
-- Using CATKIN DEVEL_PREFIX: /home/moquitas/ros_ws/devel  
-- Using CMAKE_PREFIX_PATH: /home/moquitas/ros_ws/devel:/home/moquitas/hd_map_ws/devel:/opt/ros/kinetic  
-- This workspace overlays: /home/moquitas/ros_ws/devel:/home/moquitas/hd_map_ws/devel:/opt/ros/kinetic  
-- Using PYTHON_EXECUTABLE: /usr/bin/python  
-- Using Debian Python package layout  
-- Using empy: /usr/bin/empy  
-- Using CATKIN_ENABLE_TESTING: ON  
-- Call enable_testing()  
-- Using CATKIN_TEST_RESULTS_DIR: /home/moquitas/ros_ws/build/test_results  
-- Found gtest sources under '/usr/src/gtest': gtests will be built  
-- Using Python nosetests: /usr/bin/nosetests-2.7  
-- catkin 0.7.8  
-- BUILD_SHARED_LIBS is on  
--  
-- traversing 1 packages in topological order:  
-- - beginner_tutorials  
--  
-- +++ processing catkin package: 'beginner_tutorials'  
-- == add subdirectory(beginner_tutorials)  
-- Using these message generators: gencpp;geneus;genlisp;gennodejs;genpy  
-- beginner_tutorials: 1 messages, 1 services  
-- Configuring done  
-- Generating done  
-- Build files have been written to: /home/moquitas/ros_ws/build  
####  
Running command: "make -j8 -l8" in "/home/moquitas/ros_ws/build"  
####  
Scanning dependencies of target beginner_tutorials_generate_messages_check_deps_AddTwoInts  
[ 0%] Built target std_msgs_generate_messages_eus  
[ 0%] Built target std_msgs_generate_messages_cpp  
[ 6%] Built target std_msgs_generate_messages_nodejs  
[ 6%] Built target std_msgs_generate_messages_py  
[ 13%] Built target beginner_tutorials_node  
[ 13%] Built target std_msgs_generate_messages_lisp  
[ 13%] Built target beginner_tutorials_generate_messages_check_deps_Num  
[ 13%] Built target beginner_tutorials_generate_messages_check_deps_AddTwoInts  
[ 20%] Generating C++ code from beginner_tutorials/AddTwoInts.srv  
[ 20%] Generating Javascript code from beginner_tutorials/AddTwoInts.srv  
[ 33%] Generating Python code from SRV beginner_tutorials/AddTwoInts  
[ 40%] Generating Euslisp code from beginner_tutorials/AddTwoInts.srv  
[ 46%] Generating Lisp code from beginner_tutorials/AddTwoInts.srv  
[ 53%] Built target beginner_tutorials_generate_messages_nodejs  
[ 60%] Built target beginner_tutorials_generate_messages_lisp  
[ 73%] Built target beginner_tutorials_generate_messages_eus  
[ 80%] Generating Python msg _init_.py for beginner_tutorials  
[ 86%] Generating Python srv _init_.py for beginner_tutorials  
[ 93%] Built target beginner_tutorials_generate_messages_cpp  
[ 100%] Built target beginner_tutorials_generate_messages_py  
[ 100%] Built target beginner_tutorials_generate_messages  
moquitas@moquitas:~/ros_ws$
```

Uygulama 5: Publisher-Subscriber Uygulaması – 1/5

C++ Uygulaması – 1/2

1. beginner_tutorials altındaki src klasörüne gidelim.

`roscd beginner_tutorials/src`

2. Publisher dosyamızı oluşturalım.

`gedit talker.cpp`

```
talker.cpp (~/.ros_ws/src/beginner_tutorials/src) - gedit
#include "ros/ros.h"
#include "std_msgs/String.h"

#include <sstream>

/**
 * This tutorial demonstrates simple sending of messages over the ROS system.
 */
int main(int argc, char **argv)
{
    ros::init(argc, argv, "talker"); ROS'u başlatır. Düğüm adı talker olacak şekilde ayarlandı.

    ros::NodeHandle n; Tanıtıcı oluşturur.

    ros::Publisher chatter_pub = n.advertise<std_msgs::String>("chatter", 1000); Master'a chatter konusunda std_msgs/String türünde bir mesaj yayımlayacağımızı bildirir. İkinci argüman ise yayım kuyruğunun büyüklüğüdür.

    ros::Rate loop_rate(10); Döngü frekansı 10 Hz olarak ayarlandı. Saniyede 10 defa mesaj yazdırıyoruz

    int count = 0;
    while (ros::ok())
    {
        std_msgs::String msg;
        std::stringstream ss;
        ss << "hello world " << count; Mesaj oluşturuluyor
        msg.data = ss.str();

        ROS_INFO("%s", msg.data.c_str()); ROS hata verene kadar işlemler dönüyor
        Mesaj ekrana bastırılıyor

        chatter_pub.publish(msg); Mesaj yayınlanıyor

        ros::spinOnce(); Call-back için gereklidir

        loop_rate.sleep();
        ++count;
    }

    return 0;
}
```

3. Subscriber dosyamızı oluşturalım.

`gedit listener.cpp`

```
listener.cpp (~/.ros_ws/src/beginner_tutorials/src) - gedit
#include "ros/ros.h"
#include "std_msgs/String.h"

/**
 * This tutorial demonstrates simple receipt of messages over the ROS system.
 */
void chatterCallback(const std_msgs::String::ConstPtr& msg)
{
    ROS_INFO("I heard: [%s]", msg->data.c_str()); Alınan mesaj ekrana bastırılıyor

}

int main(int argc, char **argv)
{
    ros::init(argc, argv, "listener"); ROS'u başlatır. Düğüm adı listener olacak şekilde ayarlandı.

    ros::NodeHandle n; Tanıtıcı oluşturur.

    ros::Subscriber sub = n.subscribe("chatter", 1000, chatterCallback); talker a chatter konusunda std_msgs/String abone olur. Her mesaj yayınlandığında chatterCallback çağrılır. İkinci argüman ise yayım kuyruğunun büyüklüğüdür.

    ros::spin(); Bu kod bir döngü girer, mesaj geri çağırımlarını mümkün olduğu kadar hızlı çağırır.

    return 0;
}
```


Uygulama 5: Publisher-Subscriber Uygulaması – 2/5

C++ Uygulaması – 2/2

4. CMakeLists.txt dosyamızı düzenleyelim.

roscd beginner_tutorials

gedit CMakeLists.txt

```
CMakeLists.txt (~/.ros_ws/src/beginner_tutorials) - gedit
Open Save

# add_dependencies(${PROJECT_NAME} ${${PROJECT_NAME}_EXPORTED_TARGETS} ${catkin_EXPORTED_TARGETS})

## Declare a C++ executable
## With catkin_make all packages are built within a single CMake context
## The recommended prefix ensures that target names across packages don't collide
# add_executable(${PROJECT_NAME}_node src/beginner_tutorials_node.cpp)
add_executable(beginner_tutorials_node src/first_script.cpp)
add_executable(talker src/talker.cpp)
target_link_libraries(talker ${catkin_LIBRARIES})
add_dependencies(talker beginner_tutorials_generate_messages_cpp)

## Rename C++ executable without prefix
## The above recommended prefix causes long target names, the following renames the
## target back to the shorter version for ease of user use
## e.g. "roscd someones_pkg node" instead of "roscd someones_pkg someones_pkg_node"
# set_target_properties(${PROJECT_NAME}_node PROPERTIES OUTPUT_NAME node PREFIX "")

## Add cmake target dependencies of the executable
## same as for the library above
# add_dependencies(${PROJECT_NAME}_node ${${PROJECT_NAME}_EXPORTED_TARGETS} ${catkin_EXPORTED_TARGETS})
)

add_executable(listener src/listener.cpp)
target_link_libraries(listener ${catkin_LIBRARIES})
add_dependencies(listener beginner_tutorials_generate_messages_cpp)

## Specify libraries to link a library or executable target against
target_link_libraries(beginner_tutorials_node
  ${catkin_LIBRARIES}
)
```

5. Çalışma alanını derleyin.

cd ~/.ros_ws

catkin_make

```
moguztas@moguztas:~/ros_ws$ catkin_make
Base path: /home/moguztas/ros_ws
Source space: /home/moguztas/ros_ws/src
Build space: /home/moguztas/ros_ws/build
Devel space: /home/moguztas/ros_ws/devel
Install space: /home/moguztas/ros_ws/install
####
#### Running command: "make cmake_check_build_system" in "/home/moguztas/ros_ws/build"
####
-- Using CATKIN_DEVEL_PREFIX: /home/moguztas/ros_ws/devel
-- Using CMAKE_PREFIX_PATH: /home/moguztas/ros_ws/devel:/home/moguztas/hd_map_ws/devel:/opt/ros/kinetic
-- This workspace overlays: /home/moguztas/ros_ws/devel:/home/moguztas/hd_map_ws/devel:/opt/ros/kinetic
-- Using PYTHON_EXECUTABLE: /usr/bin/python
-- Using Debian Python package layout
-- Using empy: /usr/bin/empy
-- Using CATKIN_ENABLE_TESTING: ON
-- Call enable_testing()
-- Using CATKIN_TEST_RESULTS_DIR: /home/moguztas/ros_ws/build/test_results
-- Found gtest sources under '/usr/src/gtest': gtests will be built
-- Using Python nosetests: /usr/bin/nosetests-2.7
-- catkin 0.7.8
-- BUILD_SHARED_LIBS is on
-- 
-- traversing 1 packages in topological order:
--   - beginner_tutorials
-- 
-- *** processing catkin package: 'beginner_tutorials'
-- == add subdirectory(beginner_tutorials)
-- Using these message generators: gencpp;geneus;genlisp;gennodejs;genpy
-- beginner_tutorials: 1 messages, 1 services
-- Configuring done
-- Generating done
-- Build files have been written to: /home/moguztas/ros_ws/build
####
#### Running command: "make -j8 -l8" in "/home/moguztas/ros_ws/build"
####
[ 0%] Built target std_msgs_generate_messages_eus
[ 0%] Built target std_msgs_generate_messages_cpp
[ 10%] Built target beginner_tutorials_node
[ 10%] Built target std_msgs_generate_messages_nodejs
[ 10%] Built target std_msgs_generate_messages_py
[ 10%] Built target std_msgs_generate_messages_lisp
[ 10%] Built target beginner_tutorials_generate_messages_check_deps_Num
[ 10%] Built target _beginner_tutorials_generate_messages_check_deps_AddTwoInts
[ 26%] Built target beginner_tutorials_generate_messages_eus
[ 36%] Built target beginner_tutorials_generate_messages_nodejs
[ 52%] Built target beginner_tutorials_generate_messages_lisp
[ 57%] Built target beginner_tutorials_generate_messages_cpp
[ 78%] Built target beginner_tutorials_generate_messages_py
Scanning dependencies of target talker
[ 89%] Building CXX object beginner_tutorials/CMakeFiles/talker.dir/src/talker.cpp.o
[ 89%] Building CXX object beginner_tutorials/CMakeFiles/listener.dir/src/listener.cpp.o
[ 94%] Linking CXX executable /home/moguztas/ros_ws/devel/lib/beginner_tutorials/talker
[100%] Linking CXX executable /home/moguztas/ros_ws/devel/lib/beginner_tutorials/listener
[100%] Built target talker
[100%] Built target listener
moguztas@moguztas:~/ros_ws$
```

Uygulama 5: Publisher-Subscriber Uygulaması – 3/5

Python Uygulaması – 1/2

1. beginner_tutorials klasörüne gidip scripts klasörü oluşturalım.

```
roscd beginner_tutorials/src
```

```
mkdir scripts
```

```
cd scripts
```

2. Publisher dosyamızı oluşturalım.

```
gedit talker.py
```

```
#!/usr/bin/env python
## Simple talker demo

import rospy
from std_msgs.msg import String

def talker():
    pub = rospy.Publisher('chatter', String, queue_size=10)
    rospy.init_node('talker', anonymous=True)
    rate = rospy.Rate(10) # 10hz
    while not rospy.is_shutdown():
        hello_str = "hello world %s" % rospy.get_time()
        rospy.loginfo(hello_str)
        pub.publish(hello_str)
        rate.sleep()

if __name__ == '__main__':
    try:
        talker()
    except rospy.ROSInterruptException:
        pass
```

Master'a **chatter** konusunda *std_msgs/String* türünde bir mesaj yayımlayacağımızı bildirir. Üçüncü argüman ise yayım kuyruğunun büyüklüğüdür.

talker düğümü oluşturuluyor

Döngü frekansı 10 Hz olarak ayarlanıyor.

rospy.get_time() Mesaj oluşturuluyor

rospy.loginfo(hello_str) Mesaj ekrana bastırılıyor

pub.publish(hello_str) Mesaj yayınlanıyor.

rate.sleep() ROS hata verene kadar işlemler dönüyor

3. Subscriber dosyamızı oluşturalım.

```
gedit listener.py
```

```
#!/usr/bin/env python
## Simple listener demo

import rospy
from std_msgs.msg import String

def callback(data):
    rospy.loginfo(rospy.get_caller_id() + 'I heard %s', data.data)

def listener():
    # In ROS, nodes are uniquely named. If two nodes with the same
    # name are launched, the previous one is kicked off. The
    # anonymous=True flag means that rospy will choose a unique
    # name for our 'listener' node so that multiple listeners can
    # run simultaneously.
    rospy.init_node('listener', anonymous=True)

    rospy.Subscriber('chatter', String, callback)

    # spin() simply keeps python from exiting until this node is stopped
    rospy.spin()

if __name__ == '__main__':
    listener()
```

callback fonksiyonu

rospy.loginfo(rospy.get_caller_id() + 'I heard %s', data.data) Alınan mesaj ekrana bastırılıyor

listener düğümü oluşturuluyor.

Konuya abone olunuyor.

rospy.spin() Düğüm sonlanan kadar işlemler devam ediyor.

Uygulama 5: Publisher-Subscriber Uygulaması – 4/5

Python Uygulaması – 2/2

4. Dosyalarımızı çalıştırılabilir yapalım.

ls

```
moguztas@moguztas:~/ros_ws/src/beginner_tutorials/scripts$ ls
listener.py talker.py
moguztas@moguztas:~/ros_ws/src/beginner_tutorials/scripts$
```

chmod +x listener.py

```
moguztas@moguztas:~/ros_ws/src/beginner_tutorials/scripts$ chmod +x listener.py
moguztas@moguztas:~/ros_ws/src/beginner_tutorials/scripts$ ls
listener.py talker.py
moguztas@moguztas:~/ros_ws/src/beginner_tutorials/scripts$
```

5. Çalışma alanını derleyin.

cd ~/ros_ws

catkin_make

```
moguztas@moguztas:~/ros_ws$ catkin_make
Base path: /home/moguztas/ros_ws
Source space: /home/moguztas/ros_ws/src
Build space: /home/moguztas/ros_ws/build
Devel space: /home/moguztas/ros_ws/devel
Install space: /home/moguztas/ros_ws/install
####
#### Running command: "make cmake_check_build_system" in "/home/moguztas/ros_ws/build"
####
-- Using CATKIN_DEVEL_PREFIX: /home/moguztas/ros_ws/devel
-- Using CMAKE_PREFIX_PATH: /home/moguztas/ros_ws/devel;/home/moguztas/hd_map_ws/devel;/opt/ros/kinetic
-- This workspace overlays: /home/moguztas/ros_ws/devel;/home/moguztas/hd_map_ws/devel;/opt/ros/kinetic
-- Using PYTHON_EXECUTABLE: /usr/bin/python
-- Using Debian Python package layout
-- Using empy: /usr/bin/empy
-- Using CATKIN_ENABLE_TESTING: ON
-- Call enable_testing()
-- Using CATKIN_TEST_RESULTS_DIR: /home/moguztas/ros_ws/build/test_results
-- Found gtest sources under "/usr/src/gtest": gtests will be built
-- Using Python nosetests: /usr/bin/nosetests-2.7
-- catkin 0.7.8
-- BUILD_SHARED_LIBS is on
-- 
-- traversing 1 packages in topological order:
-- ~ - beginner_tutorials
-- 
-- ++ processing catkin package: 'beginner_tutorials'
-- ==> add subdirectory(beginner_tutorials)
-- Using these message generators: gencpp;geneus;genlisp;gennodejs;genpy
-- beginner_tutorials: 1 messages, 1 services
-- Configuring done
-- Generating done
-- Build files have been written to: /home/moguztas/ros_ws/build
####
#### Running command: "make -j8 -l8" in "/home/moguztas/ros_ws/build"
####
[ 0%] Built target std_msgs_generate_messages_eus
[ 0%] Built target std_msgs_generate_messages_cpp
[ 10%] Built target beginner_tutorials_node
[ 10%] Built target std_msgs_generate_messages_nodejs
[ 10%] Built target std_msgs_generate_messages_py
[ 10%] Built target std_msgs_generate_messages_lisp
[ 10%] Built target beginner_tutorials_generate_messages_check_deps_AddTwoInts
[ 10%] Built target beginner_tutorials_generate_messages_check_deps_AddTwoInts
[ 26%] Built target beginner_tutorials_generate_messages_eus
[ 36%] Built target beginner_tutorials_generate_messages_nodejs
[ 52%] Built target beginner_tutorials_generate_messages_lisp
[ 57%] Built target beginner_tutorials_generate_messages_cpp
[ 78%] Built target beginner_tutorials_generate_messages_py
Scanning dependencies of target talker
Scanning dependencies of target listener
[ 78%] Built target beginner_tutorials_generate_messages
[ 89%] Building CXX object beginner_tutorials/CMakeFiles/talker.dir/src/talker.cpp.o
[ 89%] Building CXX object beginner_tutorials/CMakeFiles/listener.dir/src/listener.cpp.o
[ 94%] Linking CXX executable /home/moguztas/ros_ws/devel/lib/beginner_tutorials/talker
[100%] Linking CXX executable /home/moguztas/ros_ws/devel/lib/beginner_tutorials/listener
[100%] Built target talker
[100%] Built target listener
moguztas@moguztas:~/ros_ws$
```

Uygulama 5: Publisher-Subscriber Uygulaması – 5/5

6. Uygulamayı çalıştırma

6.1. Terminal 1: *roscore*

6.2. Terminal 2:

- (C++) *roslaunch beginner_tutorials talker*
- (Python) *roslaunch beginner_tutorials talker.py*

```
moguztas@moguztas:~/ros_ws$ roslaunch beginner_tutorials talker
[ INFO] [1567209367.884622634]: hello world 0
[ INFO] [1567209367.984790365]: hello world 1
[ INFO] [1567209368.084775385]: hello world 2
[ INFO] [1567209368.184757122]: hello world 3
[ INFO] [1567209368.284752723]: hello world 4
[ INFO] [1567209368.384781284]: hello world 5
[ INFO] [1567209368.484788813]: hello world 6
[ INFO] [1567209368.584788005]: hello world 7
[ INFO] [1567209368.684791028]: hello world 8
[ INFO] [1567209368.784744496]: hello world 9
[ INFO] [1567209368.884766553]: hello world 10
[ INFO] [1567209368.984751599]: hello world 11
[ INFO] [1567209369.084771260]: hello world 12
[ INFO] [1567209369.184755739]: hello world 13
[ INFO] [1567209369.284732317]: hello world 14
[ INFO] [1567209369.384752488]: hello world 15
[ INFO] [1567209369.484772846]: hello world 16
[ INFO] [1567209369.584798485]: hello world 17
[ INFO] [1567209369.684763541]: hello world 18
[ INFO] [1567209369.784781674]: hello world 19
[ INFO] [1567209369.884801013]: hello world 20
[ INFO] [1567209369.984798947]: hello world 21
[ INFO] [1567209370.084707976]: hello world 22
[ INFO] [1567209370.184756887]: hello world 23
[ INFO] [1567209370.284791587]: hello world 24
[ INFO] [1567209370.384794507]: hello world 25
```

6.3. Terminal 3:

- (C++) *roslaunch beginner_tutorials listener*
- (Python) *roslaunch beginner_tutorials listener.py*

```
moguztas@moguztas:~/ros_ws$ roslaunch beginner_tutorials listener.py
[INFO] [1567209575.599204]: /listener_16717_1567209575367I heard hello world 165
[INFO] [1567209575.699208]: /listener_16717_1567209575367I heard hello world 166
[INFO] [1567209575.799234]: /listener_16717_1567209575367I heard hello world 167
[INFO] [1567209575.899221]: /listener_16717_1567209575367I heard hello world 168
[INFO] [1567209575.999226]: /listener_16717_1567209575367I heard hello world 169
[INFO] [1567209576.099204]: /listener_16717_1567209575367I heard hello world 170
[INFO] [1567209576.199212]: /listener_16717_1567209575367I heard hello world 171
[INFO] [1567209576.299228]: /listener_16717_1567209575367I heard hello world 172
[INFO] [1567209576.399248]: /listener_16717_1567209575367I heard hello world 173
[INFO] [1567209576.499223]: /listener_16717_1567209575367I heard hello world 174
[INFO] [1567209576.599137]: /listener_16717_1567209575367I heard hello world 175
[INFO] [1567209576.699225]: /listener_16717_1567209575367I heard hello world 176
[INFO] [1567209576.799226]: /listener_16717_1567209575367I heard hello world 177
[INFO] [1567209576.899234]: /listener_16717_1567209575367I heard hello world 178
[INFO] [1567209576.999163]: /listener_16717_1567209575367I heard hello world 179
[INFO] [1567209577.099210]: /listener_16717_1567209575367I heard hello world 180
[INFO] [1567209577.199122]: /listener_16717_1567209575367I heard hello world 181
[INFO] [1567209577.299158]: /listener_16717_1567209575367I heard hello world 182
[INFO] [1567209577.399200]: /listener_16717_1567209575367I heard hello world 183
[INFO] [1567209577.499195]: /listener_16717_1567209575367I heard hello world 184
[INFO] [1567209577.599238]: /listener_16717_1567209575367I heard hello world 185
[INFO] [1567209577.699100]: /listener_16717_1567209575367I heard hello world 186
[INFO] [1567209577.799165]: /listener_16717_1567209575367I heard hello world 187
[INFO] [1567209577.899168]: /listener_16717_1567209575367I heard hello world 188
[INFO] [1567209577.999260]: /listener_16717_1567209575367I heard hello world 189
[INFO] [1567209578.099261]: /listener_16717_1567209575367I heard hello world 190
```

Uygulama 6: Service-Client Uygulaması – 1/5

C++ Uygulaması – 1/2

1. beginner_tutorials altındaki src klasörüne gidelim.

roscd beginner_tutorials/src

2. Server dosyamızı oluşturalım.

gedit add_two_ints_server.cpp

```
add_two_ints_server.cpp (~/.ros_ws/src/beginner_tutorials/src) - gedit
Open Save
#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;
}
```

3. Client dosyamızı oluşturalım.

gedit add_two_ints_client.cpp

```
add_two_ints_client.cpp (~/.ros_ws/src/beginner_tutorials/src) - gedit
Open Save
#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;
    }

    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;
}
```

Uygulama 6: Service-Client Uygulaması – 2/5

C++ Uygulaması – 2/2

4. CMakeLists.txt dosyamızı düzenleyelim.

roscd beginner_tutorials

gedit CMakeLists.txt

```
CMakeLists.txt (~/.ros_ws/src/beginner_tutorials) - gedit
Open Save

## Add cmake target dependencies of the library
## as an example, code may need to be generated before libraries
## either from message generation or dynamic reconfigure
add_dependencies(${PROJECT_NAME} ${${PROJECT_NAME}_EXPORTED_TARGETS} ${catkin_EXPORTED_TARGETS})

## Declare a C++ executable
## With catkin_make all packages are built within a single CMake context
## The recommended prefix ensures that target names across packages don't collide
add_executable(${PROJECT_NAME}_node src/beginner_tutorials_node.cpp)
add_executable(beginner_tutorials_node src/first_script.cpp)
add_executable(talker src/talker.cpp)
target_link_libraries(talker ${catkin_LIBRARIES})
add_dependencies(talker beginner_tutorials_generate_messages_cpp)

## Rename C++ executable without prefix
## The above recommended prefix causes long target names, the following renames the
## target back to the shorter version for ease of user use
## e.g. "roslaunch someones_pkg node" instead of "roslaunch someones_pkg someones_pkg_node"
set_target_properties(${PROJECT_NAME}_node PROPERTIES OUTPUT_NAME PREFIX "")

## Add cmake target dependencies of the executable
## same as for the library above
add_dependencies(${PROJECT_NAME}_node ${${PROJECT_NAME}_EXPORTED_TARGETS} ${catkin_EXPORTED_TARGETS})

add_executable(listener src/listener.cpp)
target_link_libraries(listener ${catkin_LIBRARIES})
add_dependencies(listener beginner_tutorials_generate_messages_cpp)

add_executable(add_two_ints_server src/add_two_ints_server.cpp)
target_link_libraries(add_two_ints_server ${catkin_LIBRARIES})
add_dependencies(add_two_ints_server beginner_tutorials_gencpp)

add_executable(add_two_ints_client src/add_two_ints_client.cpp)
target_link_libraries(add_two_ints_client ${catkin_LIBRARIES})
add_dependencies(add_two_ints_client beginner_tutorials_gencpp)

## Specify libraries to link a library or executable target against
target_link_libraries(beginner_tutorials_node
  ${catkin_LIBRARIES})

)
```

5. Çalışma alanını derleyin.

cd ~/ros_ws

catkin_make

```
moguztas@moguztas:~/ros_ws$ catkin_make
Base path: /home/moguztas/ros_ws
Source space: /home/moguztas/ros_ws/src
Build space: /home/moguztas/ros_ws/build
Devel space: /home/moguztas/ros_ws/devel
Install space: /home/moguztas/ros_ws/install
####
#### Running command: "make cmake_check_build_system" in "/home/moguztas/ros_ws/build"
####
-- Using CATKIN_DEVEL_PREFIX: /home/moguztas/ros_ws/devel
-- Using CMAKE_PREFIX_PATH: /home/moguztas/ros_ws/devel;/home/moguztas/hd_map_ws/devel;/opt/ros/kinetic
-- This workspace overlays: /home/moguztas/ros_ws/devel;/home/moguztas/hd_map_ws/devel;/opt/ros/kinetic
-- Using PYTHON_EXECUTABLE: /usr/bin/python
-- Using Debian Python package layout
-- Using empy: /usr/bin/empy
-- Using CATKIN_ENABLE_TESTING: ON
-- Call enable_testing()
-- Using CATKIN_TEST_RESULTS_DIR: /home/moguztas/ros_ws/build/test_results
-- Found gtest sources under '/usr/src/gtest': gtests will be built
-- Using Python nosetests: /usr/bin/nosetests-2.7
-- catkin 0.7.8
-- BUILD_SHARED_LIBS is on
--
-- traversing 1 packages in topological order:
-- -- beginner_tutorials
--
-- +++ processing catkin package: 'beginner_tutorials'
-- ==> add subdirectory(beginner_tutorials)
-- Using these message generators: gencpp;geneus;genlisp;gennodejs;genpy
-- beginner_tutorials: 1 messages, 1 services
-- Configuring done
-- Generating done
-- Build files have been written to: /home/moguztas/ros_ws/build
####
#### Running command: "make -j8 -l8" in "/home/moguztas/ros_ws/build"
####
[ 4%] Built target std_msgs_generate_messages_cpp
[ 8%] Built target beginner_tutorials_node
[ 8%] Built target std_msgs_generate_messages_nodejs
[ 8%] Built target std_msgs_generate_messages_eus
[ 8%] Built target std_msgs_generate_messages_py
[ 8%] Built target std_msgs_generate_messages_lisp
[ 8%] Built target beginner_tutorials_generate_messages_check_deps_AddTwoInts
[ 8%] Built target beginner_tutorials_generate_messages_check_deps_Num
[ 21%] Built target beginner_tutorials_generate_messages_eus
[ 30%] Built target beginner_tutorials_generate_messages_nodejs
[ 39%] Built target beginner_tutorials_generate_messages_cpp
[ 56%] Built target beginner_tutorials_generate_messages_py
[ 65%] Built target beginner_tutorials_generate_messages_lisp
Scanning dependencies of target beginner_tutorials_gencpp
[ 65%] Built target beginner_tutorials_gencpp
[ 65%] Built target beginner_tutorials_generate_messages
[ 73%] Built target talker
[ 82%] Built target listener
Scanning dependencies of target add_two_ints_server
[ 86%] Building CXX object beginner_tutorials/CMakeFiles/add_two_ints_server.dir/src/add_two_ints_server.cpp.o
[ 91%] Building CXX object beginner_tutorials/CMakeFiles/add_two_ints_client.dir/src/add_two_ints_client.cpp.o
[ 95%] Linking CXX executable /home/moguztas/ros_ws/devel/lib/beginner_tutorials/add_two_ints_client
[ 95%] Built target add_two_ints_client
[100%] Linking CXX executable /home/moguztas/ros_ws/devel/lib/beginner_tutorials/add_two_ints_server
[100%] Built target add_two_ints_server
moguztas@moguztas:~/ros_ws$
```

Uygulama 6: Service-Client Uygulaması – 3/5

Python Uygulaması – 1/2

1. beginner_tutorials klasörüne gidip scripts klasörü oluşturalım.

```
roscd beginner_tutorials/src
```

```
mkdir scripts
```

```
cd scripts
```

2. Server dosyamızı oluşturalım.

```
gedit add_two_ints_server.py
```

3. Client dosyamızı oluşturalım.

```
gedit add_two_ints_client.py
```

```
add_two_ints_server.py (~/.ros_ws/src/beginner_tutorials/scripts) - gedit
Open Save

#!/usr/bin/env python

from beginner_tutorials.srv import AddTwoInts, AddTwoIntsResponse
import rospy

def handle_add_two_ints(req):
    print "Returning [%s + %s = %s]"%(req.a, req.b, (req.a + req.b))
    return AddTwoIntsResponse(req.a + req.b)

def add_two_ints_server():
    rospy.init_node('add_two_ints_server')
    s = rospy.Service('add_two_ints', AddTwoInts, handle_add_two_ints)
    print "Ready to add two ints."
    rospy.spin()

if __name__ == "__main__":
    add_two_ints_server()
```

```
add_two_ints_client.py (~/.ros_ws/src/beginner_tutorials/scripts) - gedit
Open Save

#!/usr/bin/env python

import sys
import rospy
from beginner_tutorials.srv import *

def add_two_ints_client(x, y):
    rospy.wait_for_service('add_two_ints')
    try:
        add_two_ints = rospy.ServiceProxy('add_two_ints', AddTwoInts)
        resp1 = add_two_ints(x, y)
        return resp1.sum
    except rospy.ServiceException, e:
        print "Service call failed: %s"%e

def usage():
    return "%s [x y]"%sys.argv[0]

if __name__ == "__main__":
    if len(sys.argv) == 3:
        x = int(sys.argv[1])
        y = int(sys.argv[2])
    else:
        print usage()
        sys.exit(1)
    print "Requesting %s+%s"%(x, y)
    print "%s + %s = %s"%(x, y, add_two_ints_client(x, y))
```

Uygulama 6: Service-Client Uygulaması – 4/5

Python Uygulaması – 2/2

4. Dosyalarımızı çalıştırılabilir yapalım.

```
chmod +x add_two_ints_server.py
```

```
chmod +x add_two_ints_client.py
```

```
ls
```

```
moguztas@moguztas:~/ros_ws/src/beginner_tutorials/scripts$ ls
add_two_ints_client.py add_two_ints_server.py listener.py talker.py
moguztas@moguztas:~/ros_ws/src/beginner_tutorials/scripts$
```

5. Çalışma alanını derleyin.

```
cd ~/ros_ws
```

```
catkin_make
```

```
moguztas@moguztas:~/ros_ws$ catkin_make
Base path: /home/moguztas/ros_ws
Source space: /home/moguztas/ros_ws/src
Build space: /home/moguztas/ros_ws/build
Devel space: /home/moguztas/ros_ws/devel
Install space: /home/moguztas/ros_ws/install
###
### Running command: "make cmake_check_build_system" in "/home/moguztas/ros_ws/build"
###
-- Using CATKIN_DEVEL_PREFIX: /home/moguztas/ros_ws/devel
-- Using CMAKE_PREFIX_PATH: /home/moguztas/ros_ws/devel;/home/moguztas/hd_map_ws/devel;/opt/ros/kinetic
-- This workspace overlays: /home/moguztas/ros_ws/devel;/home/moguztas/hd_map_ws/devel;/opt/ros/kinetic
-- Using PYTHON_EXECUTABLE: /usr/bin/python
-- Using Debian Python package layout
-- Using empy: /usr/bin/empy
-- Using CATKIN_ENABLE_TESTING: ON
-- Call enable_testing()
-- Using CATKIN_TEST_RESULTS_DIR: /home/moguztas/ros_ws/build/test_results
-- Found gtest sources under '/usr/src/gtest': gtests will be built
-- Using Python nosetests: /usr/bin/nosetests-2.7
-- catkin 0.7.8
-- BUILD_SHARED_LIBS is on
--
-- traversing 1 packages in topological order:
-- -- beginner_tutorials
--
-- +++ processing catkin package: 'beginner_tutorials'
-- ==> add subdirectory(beginner_tutorials)
-- Using these message generators: gencpp;geneus;genlisp;gennodejs;genpy
-- beginner_tutorials: 1 messages, 1 services
-- Configuring done
-- Generating done
-- Build files have been written to: /home/moguztas/ros_ws/build
###
### Running command: "make -j8 -l8" in "/home/moguztas/ros_ws/build"
###
[ 0%] Built target std_msgs_generate_messages_eus
[ 0%] Built target std_msgs_generate_messages_cpp
[ 10%] Built target beginner_tutorials_node
[ 10%] Built target std_msgs_generate_messages_nodejs
[ 10%] Built target std_msgs_generate_messages_py
[ 10%] Built target std_msgs_generate_messages_lisp
[ 10%] Built target beginner_tutorials_generate_messages_check_deps_Num
[ 10%] Built target beginner_tutorials_generate_messages_check_deps_AddTwoInts
[ 26%] Built target beginner_tutorials_generate_messages_eus
[ 36%] Built target beginner_tutorials_generate_messages_nodejs
[ 52%] Built target beginner_tutorials_generate_messages_lisp
[ 57%] Built target beginner_tutorials_generate_messages_cpp
[ 78%] Built target beginner_tutorials_generate_messages_py
Scanning dependencies of target talker
Scanning dependencies of target listener
[ 78%] Built target beginner_tutorials_generate_messages
[ 89%] Building CXX object beginner_tutorials/CMakeFiles/talker.dir/src/talker.cpp.o
[ 89%] Building CXX object beginner_tutorials/CMakeFiles/listener.dir/src/listener.cpp.o
[ 94%] Linking CXX executable /home/moguztas/ros_ws/devel/lib/beginner_tutorials/talker
[100%] Linking CXX executable /home/moguztas/ros_ws/devel/lib/beginner_tutorials/listener
[100%] Built target talker
[100%] Built target listener
moguztas@moguztas:~/ros_ws$
```


Uygulama 6: Service-Client Uygulaması – 5/5

6. Uygulamayı çalıştırma

6.1. Terminal 1: *roscore*

6.2. Terminal 2:

- (C++) *roslaunch beginner_tutorials add_two_ints_server*
- (Python) *roslaunch beginner_tutorials add_two_ints_server.py*

```
moguztas@moguztas:~/ros_ws$ roslaunch beginner_tutorials add_two_ints_server
[ INFO] [1567210143.563009216]: Ready to add two ints.
█
```

```
moguztas@moguztas:~/ros_ws$ roslaunch beginner_tutorials add_two_ints_server
[ INFO] [1567210143.563009216]: Ready to add two ints.
[ INFO] [1567210217.638803334]: request: x=10, y=15
[ INFO] [1567210217.638825146]: sending back response: [25]
█
```

6.3. Terminal 3:

- (C++) *roslaunch beginner_tutorials add_two_ints_client 10 15*
- (Python) *roslaunch beginner_tutorials add_two_ints_client.py 10 15*

```
moguztas@moguztas:~/ros_ws$ roslaunch beginner_tutorials add_two_ints_client 10 15
[ INFO] [1567210217.638983712]: Sum: 25
moguztas@moguztas:~/ros_ws$ █
```

Uygulama 7: Verileri Kaydetme ve Oynatma

1. roscore'u çalıştıralım.

roscore

2. TurtleSim'i açalım.

roslaunch turtlesim turtlesim_node

3. Klavye kontrol düğümünü açalım.

roslaunch turtlesim turtle_teleop_key

4. beginner_tutorials klasörü altına bagfiles isimli klasör açalım.

mkdir ~/bagfiles

cd ~/bagfiles

5.1. Yayınlanan tüm konuları kaydetmek için:

roscop record -a

5.2. Bazı konuları kaydetmek için:

*roscop record -O subset /turtle1/cmd_vel
/turtle1/pose*

6. Robotumuzu klavye yardımıyla hareket ettirelim.

7. Bag dosyamızın içeriğini kontrol edelim.

roscop info bag_file



```
moguztas@moguztas:~/ros_ws/src/beginner_tutorials/bagfiles$ roscop info 2019-08-31-03-21-38.bag
path: 2019-08-31-03-21-38.bag
version: 2.0
duration: 116s (76s)
start: Aug 31 2019 03:21:38.39 (1567210898.39)
end: Aug 31 2019 03:22:54.82 (1567210974.82)
size: 662.4 KB
messages: 9552
compression: none [1/1 chunks]
types: geometry_msgs/Twist [9f195f801246fda2798d1d3e8bca84a]
       roscop_msgs/Log [acffd38cd6b6de30f128938c17c593fb]
       turtlesim/Color [353891e354491c51aeb32d673fb446]
       turtlesim/Pose [863b248d5016ca62ea2e895ae5265cf9]
topics: /rosout 4 msgs : roscop_msgs/Log (2 connections)
        /turtle1/cmd_vel 24 msgs : geometry_msgs/Twist
        /turtle1/color_sensor 4762 msgs : turtlesim/Color
        /turtle1/pose 4762 msgs : turtlesim/Pose
moguztas@moguztas:~/ros_ws/src/beginner_tutorials/bagfiles$
```

```
moguztas@moguztas:~/ros_ws/src/beginner_tutorials/bagfiles$ roscop info subset.bag
path: subset.bag
version: 2.0
duration: 35.7s
start: Aug 31 2019 03:28:08.96 (1567211288.96)
end: Aug 31 2019 03:28:44.64 (1567211324.64)
size: 178.7 KB
messages: 2253
compression: none [1/1 chunks]
types: geometry_msgs/Twist [9f195f801246fda2798d1d3e8bca84a]
       turtlesim/Pose [863b248d5016ca62ea2e895ae5265cf9]
topics: /turtle1/cmd_vel 22 msgs : geometry_msgs/Twist
        /turtle1/pose 2231 msgs : turtlesim/Pose
moguztas@moguztas:~/ros_ws/src/beginner_tutorials/bagfiles$
```

8. Bag dosyamızı oynatalım.

```
moguztas@moguztas:~/ros_ws/src/beginner_tutorials/bagfiles$ roscop play 2019-08-31-03-21-38.bag
[ INFO] [1567211137.27783777]: Opening 2019-08-31-03-21-38.bag

Waiting 0.2 seconds after advertising topics... done.

Hit space to toggle paused, or 's' to step.
[RUNNING] Bag Time: 1567210939.989698 Duration: 41.603941 / 76.436171
```

-O argümanı roscop record komutuna **subset.bag** isimli bir dosyaya sadece bu iki konuyu takip edip yazmasını söyler.

Soru & Cevap

