



# ROS入門課程

## Turtlesim操作教學

# 操作目錄

## 1 操作前的準備

ROS 資料傳輸方式	3
Topic	4
Service	5
Action	6
ROS 文件組織系統	7
catkin_ws文件設置	8
Package	9
Catkin 建構系統	10
建立 Package	11

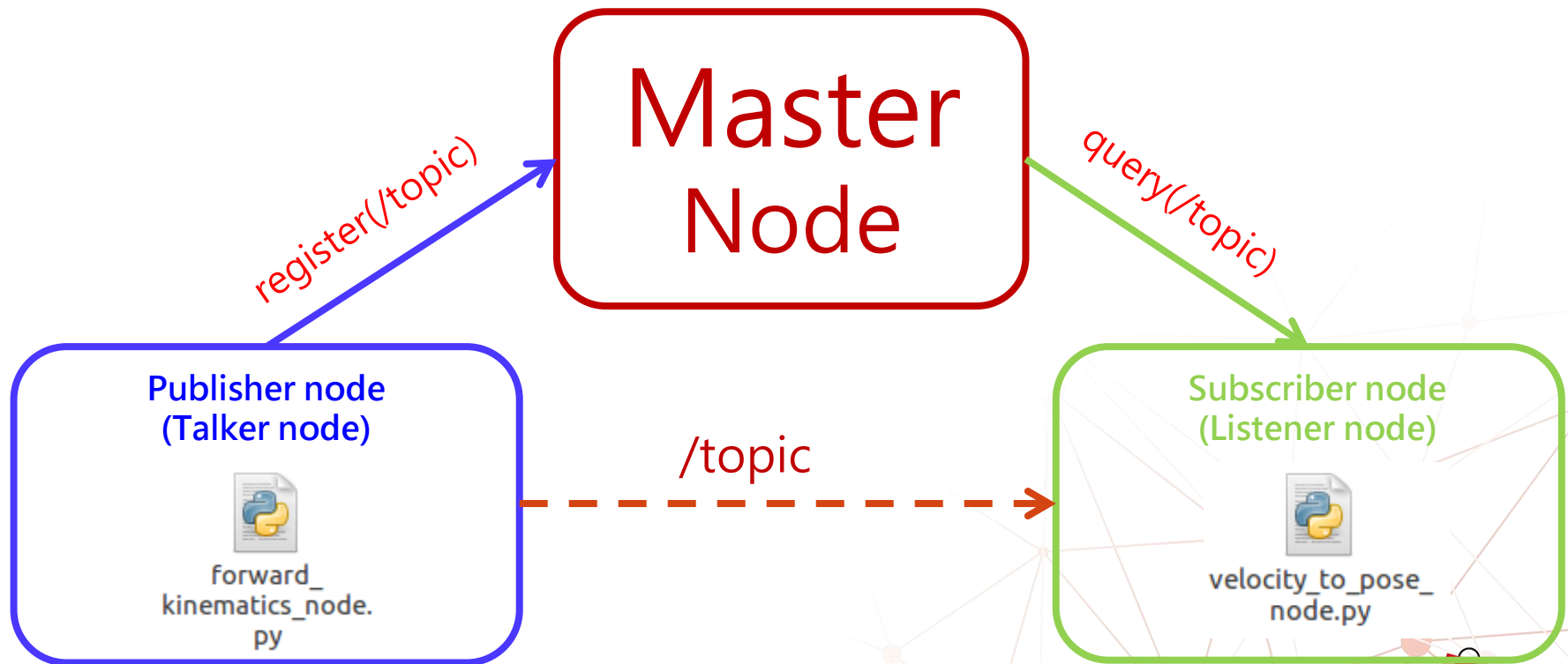
## 2 操作 Turtlesim

配置ROS工作環境	- 建立 Work space	12
	- 建立 Package	22
啟動ROS系統	- roscore	26
執行單個節點	- rosrun	29
查看節點資訊	- rosnode	32
資料視覺化	- rqt	41
查看 Topic	- rostopic	52
執行 Service	- rosservice	76
查看ROS日誌紀錄	- rqt_console	82
執行多個節點	- roslaunch	101

## 操作前的準備

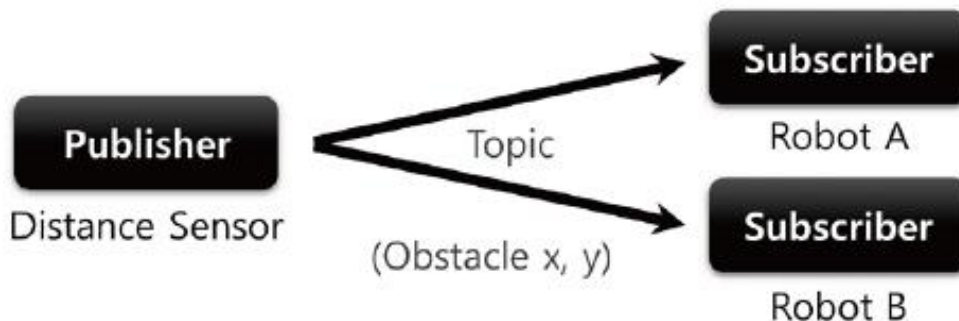
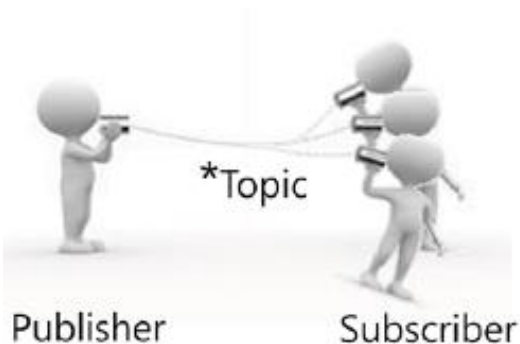
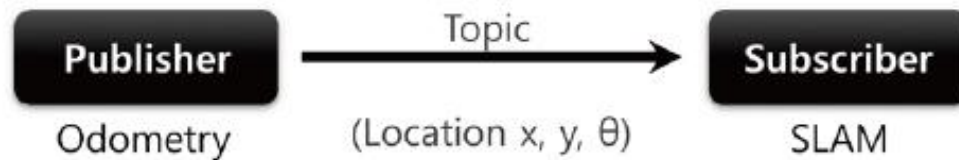
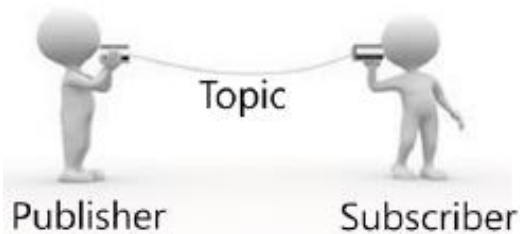
## ROS資料傳輸方式

### Node and Topic



## 操作前的準備

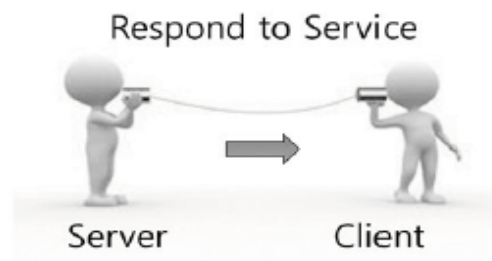
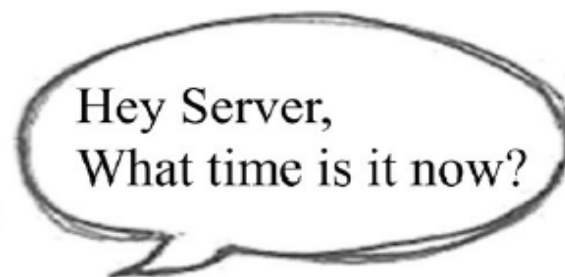
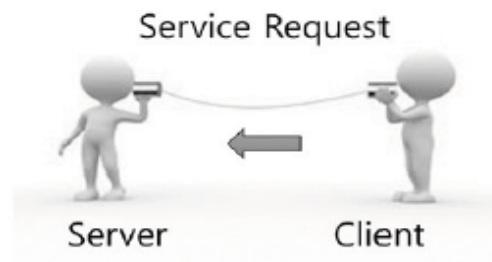
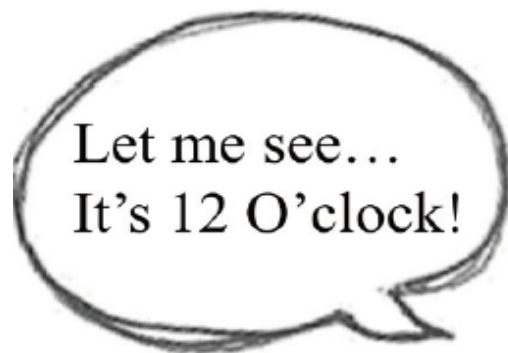
## Topic



備註：可於第 40 張 PPT 查看 Topic 的使用

## 操作前的準備

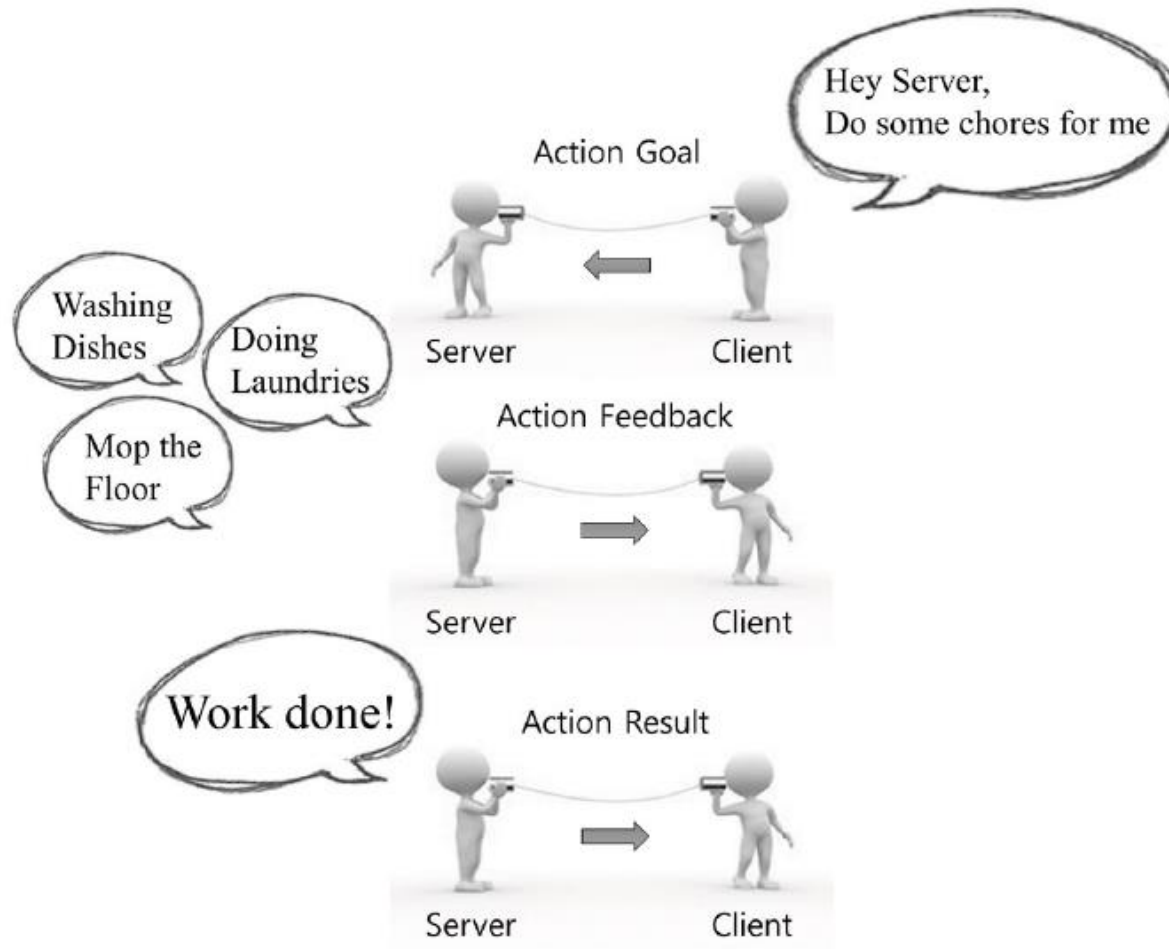
## Service



備註：可於第 75 張 PPT 查看 Service 的使用

## 操作前的準備

## Action





## 操作前的準備

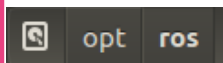
# ROS文件組織系統



安裝版本為桌面完整版

`ros-kinetic-desktop-full`

### Install folders



kinetic

roscore、rqt、RViz、  
機器人相關庫、  
導航等核心實用程式  
不常修改

### Workspace folder



catkin\_ws

可自行在  
任意目錄下建立  
詳細說明見下一頁

## 操作前的準備

## catkin\_ws文件設置

### Work space



catkin\_ws



build

利用 **catkin** 系統  
建立工作環境時  
需要的文件檔案



devel

- msg、srv的 head文件
- package的程式庫
- 可執行文件



src

package的存放區  
node的存放區



## 操作前的準備

## Package

### ROS(Robot Operating System)



Package\_1



node1.py



node2.py



node3.py



node4.py



Package\_2



node5.py



node6.py



node7.py



node8.py



Package\_3



node9.py



node10.py



node11.py



node12.py

備註：Node 不一定需要使用 python 撰寫

## 操作前的準備

## Catkin 建構系統

- CMake(Cross Platform Make)

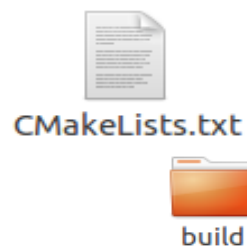
- 開放原始碼、跨平台的自動化建置系統
- 並不直接建構出最終的軟體
- 依照平台、編譯器產生標準的建構檔



ROS中，CMake修改為catkin

- catkin\_make

- 根據CMake系統所需的  
建立 ROS的工作環境，包含




## 操作前的準備


## ROS概念解析

## ROS的建構系統


## Package建立

- Catkin\_create\_pkg

- 建立 Package 的指令
- 在  目錄下執行此指令
- 自動生成 CMake系統所需的

  
CMakeLists.txt

與

  
package.xml  
CMakeLists.txt

CMake系統構建環境時所需要的描述檔案

  
package.xml

包含 Package 訊息的 Xml 格式文件

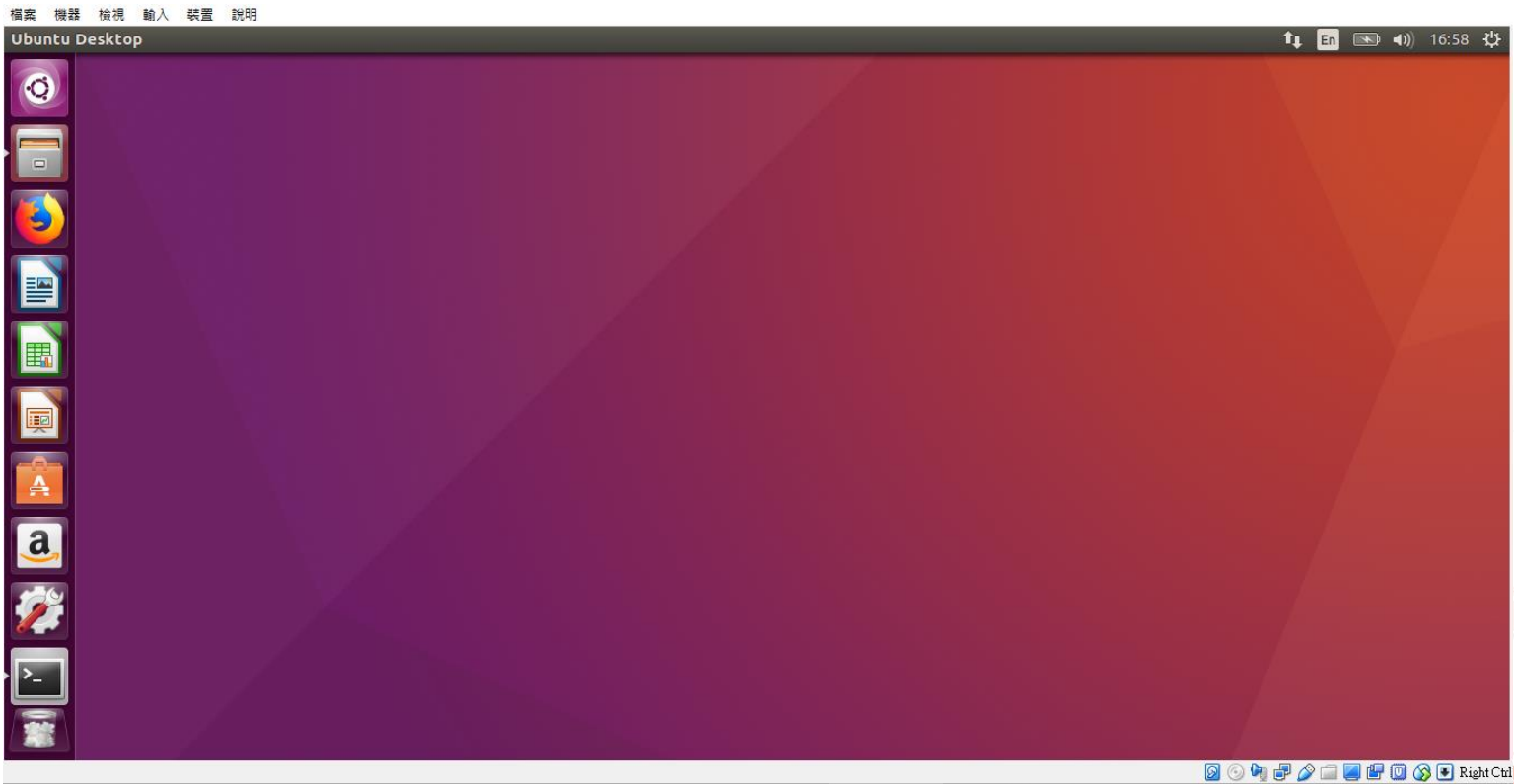


Ubuntu 16.04 LTS

## 配置ROS工作環境

## 建立Workspace folder

### 1. 利用 Virtualbox 開啟 Ubuntu 16.04 LTS

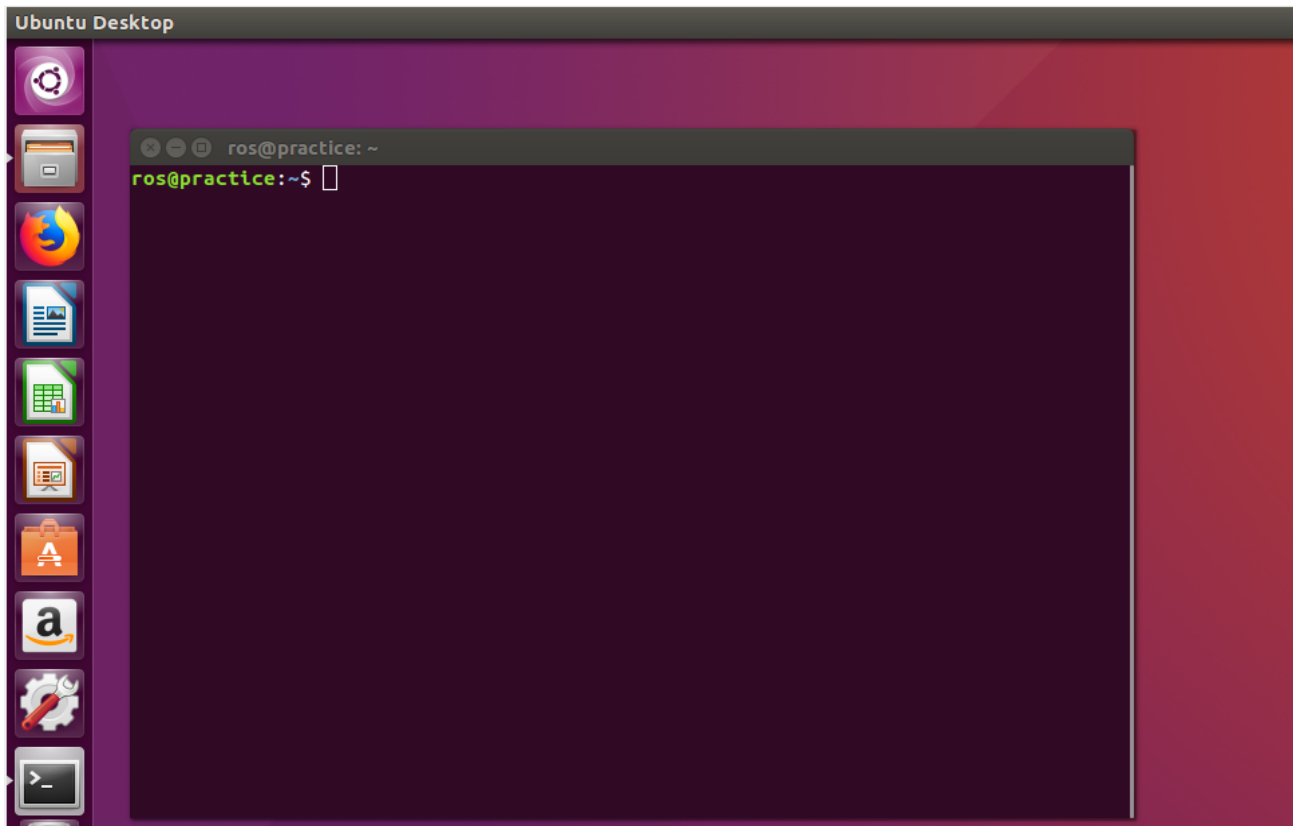




## 配置ROS工作環境

## 建立Workspace folder

### 2. 開啟 Terminal + +





## 配置ROS工作環境

## 建立Workspace folder

3. 建立



src

, 索引目錄為

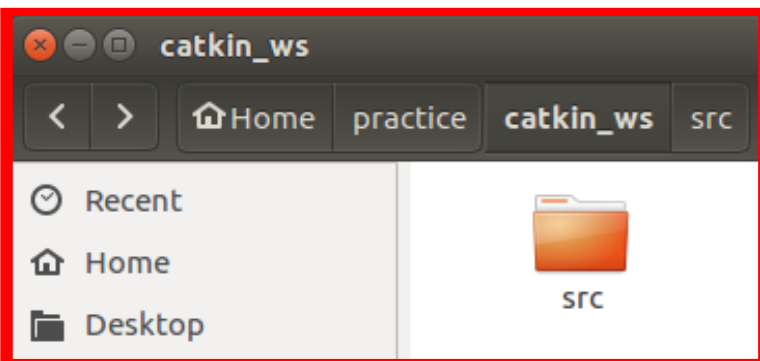
Home

practice

catkin\_ws

指令：`$ mkdir -p ~/practice/catkin_ws/src`

```
ros@practice:~$ mkdir -p ~/practice/catkin_ws/src  
ros@practice:~$
```



3-1

輸入指令

3-2

查看目錄底下是否有



src





## 配置ROS工作環境

## 建立Workspace folder

## 4.將 terminal 目錄索引至

Home

practice

catkin\_ws

指令：`$ cd ~/practice/catkin_ws/`

```
ros@practice:~$ cd ~/practice/catkin_ws/  
ros@practice:~/practice/catkin_ws$
```

4-1

輸入指令

4-2

目前 terminal  
索引的目錄



## 配置ROS工作環境

## 建立Workspace folder

5.在  建構ROS的工作環境

catkin\_ws

指令：`$ catkin_make`

```
ros@practice:~/practice/catkin_ws$ catkin make
base path: /home/ros/practice/catkin_ws
Source space: /home/ros/practice/catkin_ws/src
Build space: /home/ros/practice/catkin_ws/build
```

若無出現紅字錯誤訊息  
表示工作空間構建完成

catkin\_ws src



build



devel



src

5-1  
輸入指令

5-2  
開始建構ROS  
工作空間



## 配置ROS工作環境

## 建立Workspace folder

6.在



practice

配置ROS開發環境

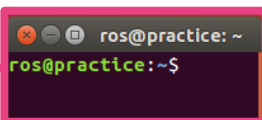
指令：`$ source ~/practice/catkin_ws/devel/setup.bash`

```
ros@practice:~/practice/catkin_ws$ source ~/practice/catkin_ws/devel/setup.bash
```

讀取setup.bash的程式碼  
設定 terminal 工作環境



setup.bash



6-1  
輸入指令



## 配置ROS工作環境

## 建立Workspace folder

## 7.配置 ROS系統的網路環境

指令：`$ sudo gedit ~/.bashrc`

```
ros@practice:~/practice/catkin_ws$ sudo gedit ~/.bashrc
```

7-1

輸入指令

使用「 gedit」

進入 Terminal

的工作環境設定



## 配置ROS工作環境

## 建立Workspace folder

## 7.配置 ROS系統的網路環境

指令：`export ROS_HOSTNAME=localhost`  
`export ROS_MASTER_URI=http://localhost:11311`

```
# enable programmable completion features (you
# this, if it's already enabled in /etc/bash.b
# sources /etc/bash.bashrc).
if ! shopt -oq posix; then
  if [ -f /usr/share/bash-completion/bash_comp
    . /usr/share/bash-completion/bash_completio
  elif [ -f /etc/bash_completion ]; then
    . /etc/bash_completion
  fi
fi
```

```
## set ROS Network ##
```

```
export ROS_HOSTNAME=localhost
export ROS_MASTER_URI=http://localhost:11311
```

7-2  
在程式碼  
最下方輸入  
指令

備註：此網路設置適用於同一台機器上使用 ROS系統



## 配置ROS工作環境

## 建立Workspace folder

## 7.配置 ROS系統的網路環境

```
## practice ##  
source /opt/ros/kinetic/setup.bash  
source ~/practice/catkin_ws/devel/setup.bash
```

```
## set ROS NetWork ##  
export ROS_HOSTNAME=localhost  
export ROS_MASTER_URI=http://localhost:11311
```

7-3

若想要 Terminal 每次都

自動載入



的工作環境，

請加入此兩行指令





## 配置ROS工作環境

## 建立Workspace folder

## 7.配置 ROS系統的網路環境

```
# ~/.bashrc: executed by bash(1) for non-login shells.
# see /usr/share/doc/bash/examples/startup-files (in the
package bash-doc)
# for examples

# If not running interactively, don't do anything
case $- in
    *i*) ;;
    *) return;;
esac

# don't put duplicate lines or lines starting with space
in the history.
# See bash(1) for more options
HISTCONTROL=ignoreboth
```

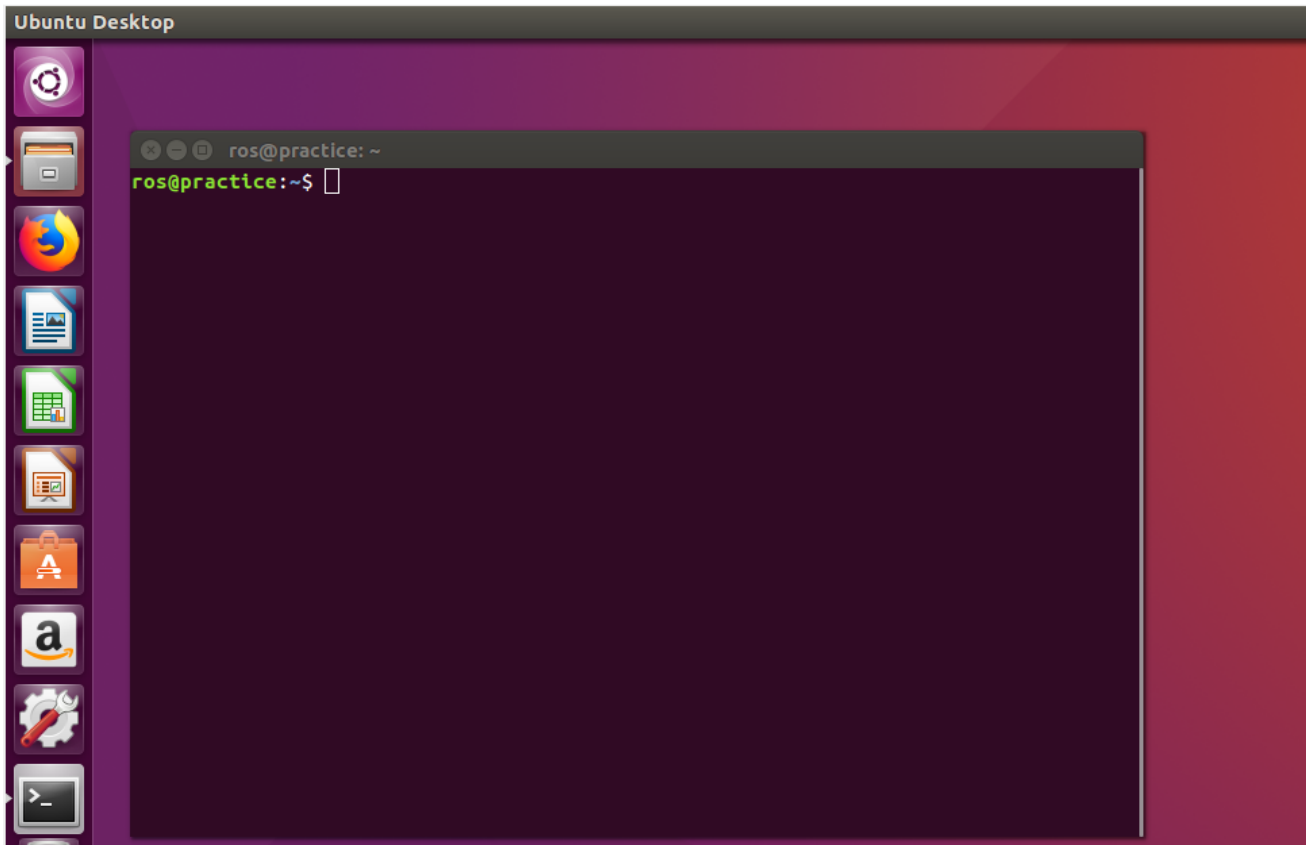
7-4  
點擊「Save」  
按鈕



## 配置ROS工作環境

## 建立Package

### 1. 開啟 Terminal + +





## 配置ROS工作環境

## 建立Package

2.將 terminal 目錄索引至

Home practice catkin\_ws src

指令：`$ cd ~/practice/catkin_ws/src`

```
ros@practice:~$ cd ~/practice/catkin_ws/src/  
ros@practice:~/practice/catkin_ws/src$
```



src

package的存放區

4-1  
輸入指令

4-2  
目前 terminal  
索引的目錄



## 配置ROS工作環境

## 建立Package

## 3.建立 Package 的描述檔案



CMakeLists.txt



package.xml

指令：`$ catkin_create_pkg first_pkg rospy std_msgs`

藍色字體：`package` 的資料夾名稱

綠色字體：`package` 需要的依賴選項(optional dependent)

```
ros@practice:~/practice/catkin_ws/src$ catkin create pkg  
first_pkg rospy std_msgs  
Created file first_pkg/package.xml  
Created file first_pkg/CMakeLists.txt  
Created folder first_pkg/src  
Successfully created files in /home/ros/practice/catkin_ws/  
src/first_pkg. Please adjust the values in package.xml.
```

3-1  
輸入指令

3-2  
建立描述  
檔案



## 配置ROS工作環境

## 建立Package

## 4.查看 Package



first\_pkg



Home

practice

catkin\_ws

src

first\_pkg



src

package的存放區  
node的存放區



CMakeLists.txt

CMake系統構建工作空間  
時所需要的描述檔案



package.xml

包含 Package 訊息的  
Xml 格式文件

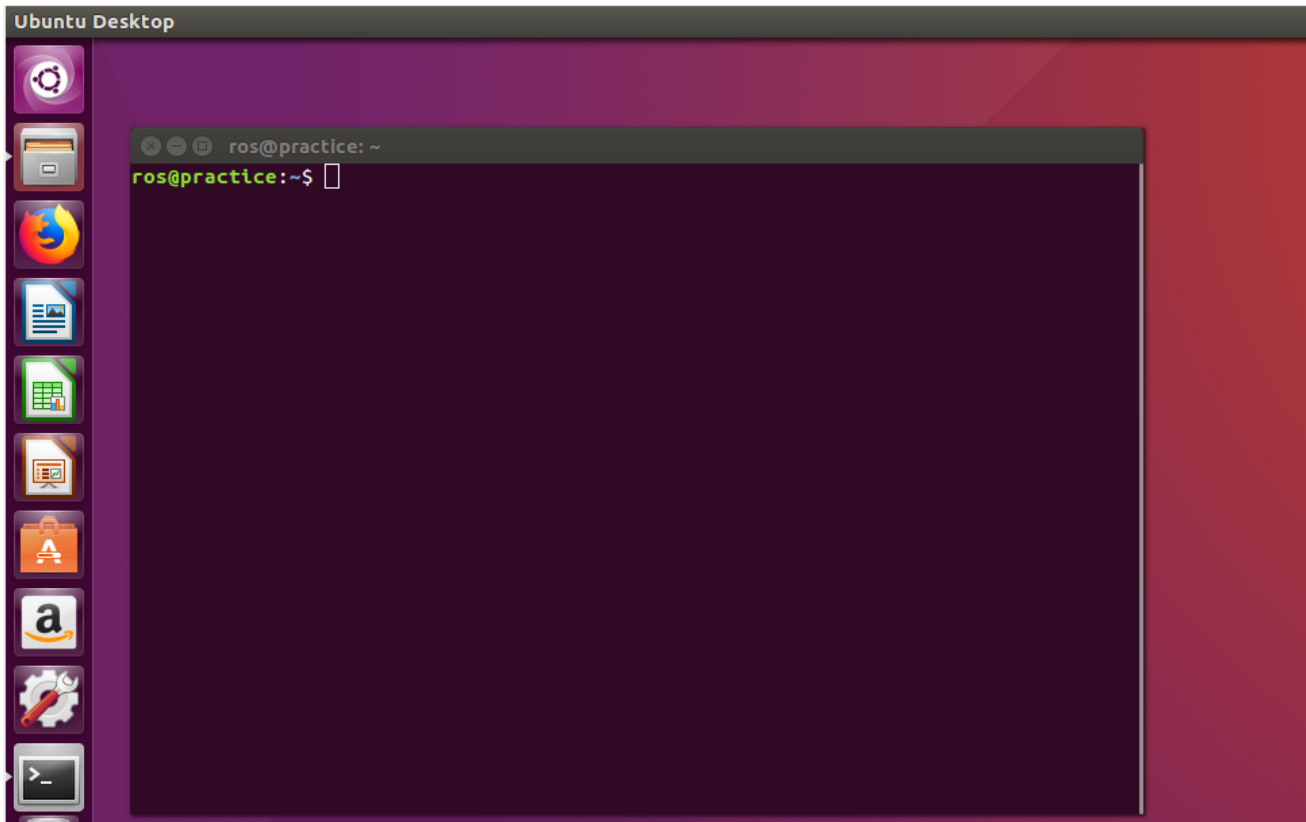




## 操作 Turtlesim

## 啟動ROS系統

### 1. 開啟 Terminal + +







操作 Turtlesim

啟動ROS系統

## 2.開啟ROS系統的 Master Node

指令 : **\$ roscore**

```
roscore http://localhost:11311/  
ros@practice:~$ roscore  
... loading to /home/ros/.ros/loa/d7f62824-22b0-11e8-9249-74da  
-practice-3711.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://localhost:42401/  
ros_comm version 1.12.12
```

2-1

輸入指令

2-2

ROS啟動  
Master  
Node



# 操作 Turtlesim

# 啟動ROS系統

## 2.開啟ROS系統的 Master Node

```
... logging to /home/ros/.ros/log/d7f62824-22b0-11e8-9249-74da38d18694/roslaunch-practice-3711.1
og
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://localhost:42401/
ros_comm version 1.12.12

SUMMARY
=====

PARAMETERS
* /rostdistro: kinetic
* /rosversion: 1.12.12

NODES
auto-starting new master
process[master]: started with pid [3721]
ROS_MASTER_URI=http://localhost:11311/

setting /run_id to d7f62824-22b0-11e8-9249-74da38d18694
process[roscout-1]: started with pid [3734]
started core service [/roscout]
```

2-3

從哪台機器啟動  
Master Node

2-4

從哪台機器啟動  
Master Node

2-4

ROS系統的Master  
在哪台機器



# 操作 Turtlesim

## 執行單個節點

3. 利用 rosrun 啟動位於

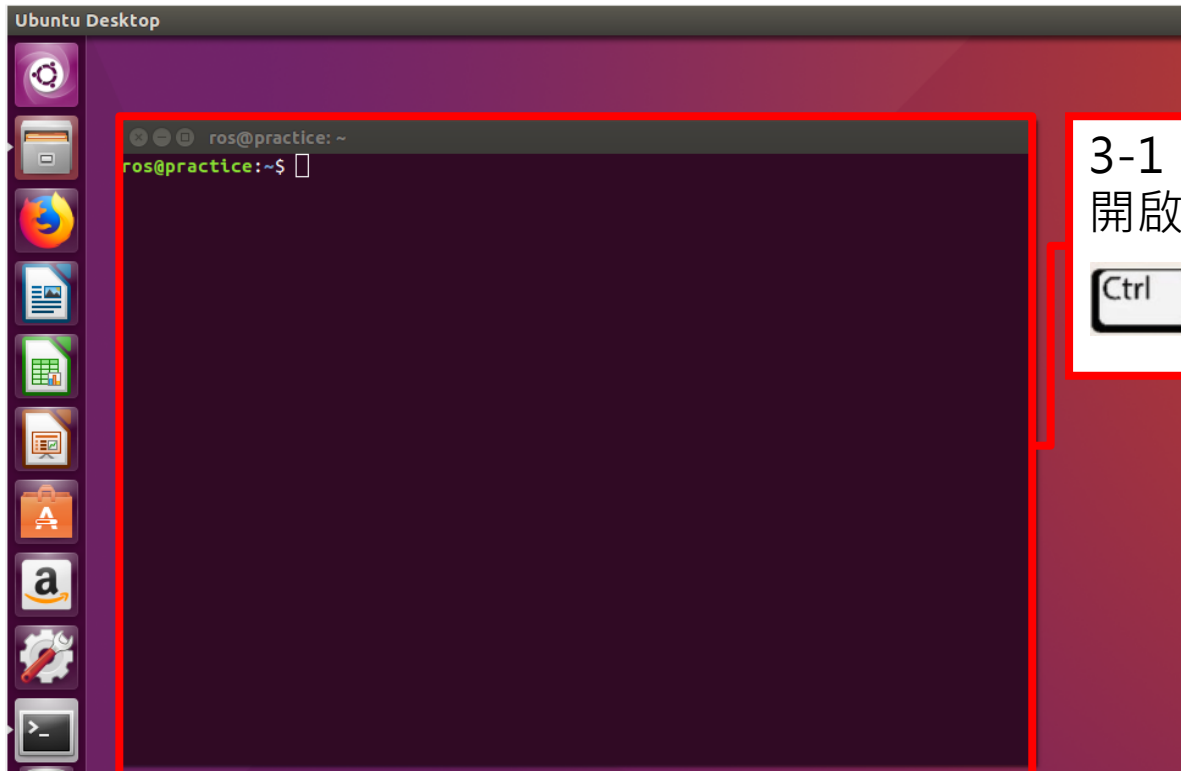


turtlesim

的

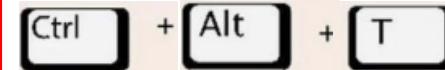


turtlesim\_node



3-1

開啟新的 Terminal





## 操作 Turtlesim

## 執行單個節點

3. 利用 rosrun 啟動位於



turtlesim

的



turtlesim\_node

指令：`$ rosrun turtlesim turtlesim_node`

藍色字體：package 的名稱

綠色字體：node 名稱

```
ros@practice:~$ rosrun turtlesim turtlesim_node
[ INFO] [1520502221.562893958]: Starting turtlesim with node name /turtlesim
[ INFO] [1520502221.575128309]: Spawning turtle [turtle1] at x=[5.544445], y=
0.000000]
```

3-2

輸入指令



## 操作 Turtlesim

## 執行單個節點

### 3.利用 rosrun 啟動位於



turtlesim

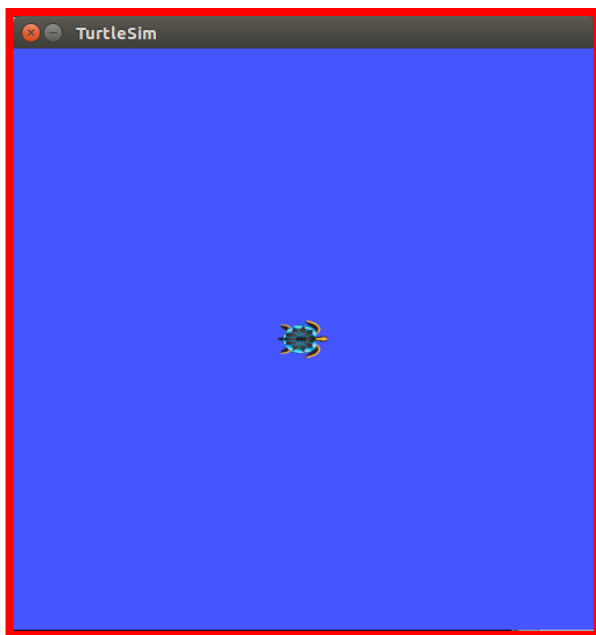
的



turtlesim\_node

```
ros@practice:~$ rosrun turtlesim turtlesim_node
```

```
[ INFO] [1520502221.562893958]: Starting turtlesim with node name /turtlesim  
[ INFO] [1520502221.575128309]: Spawning turtle [turtle1] at x=[5.544445], y=  
0.000000]
```



3-3



turtlesim\_node

啟動後資訊

3-4

啟動後畫面  
(烏龜為隨機載入)

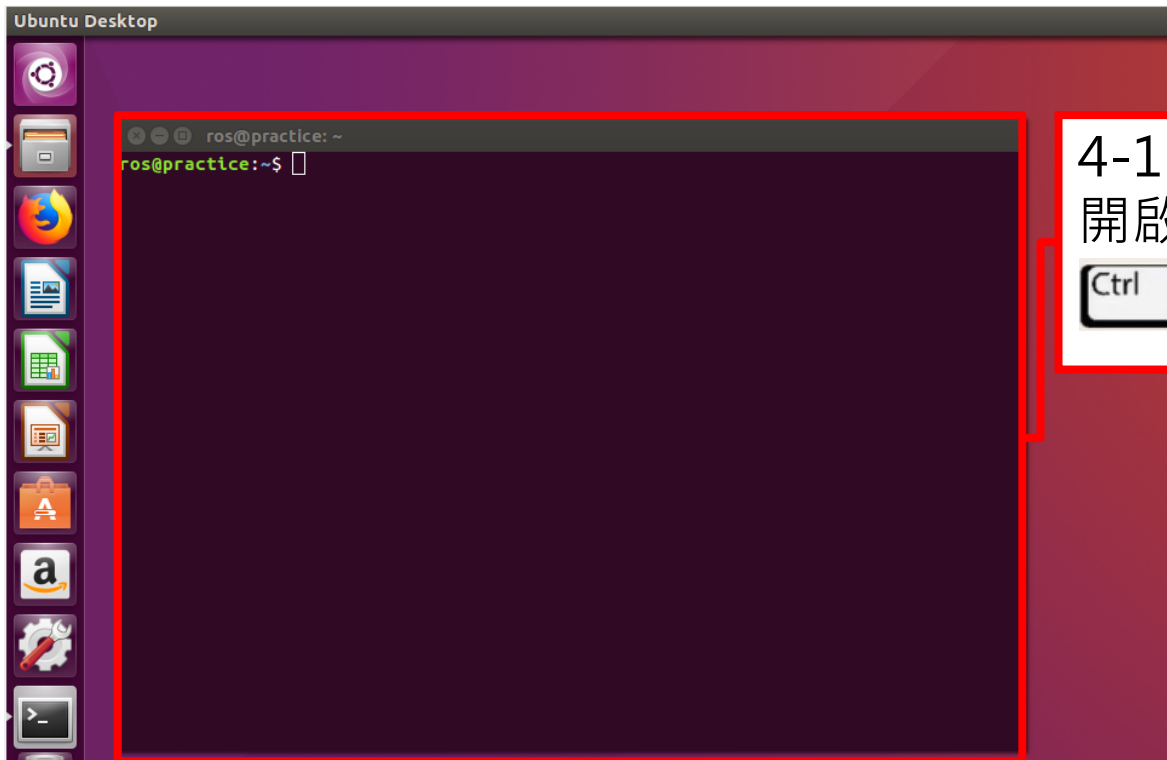




操作 Turtlesim

查看節點資訊

## 4. 利用 rosnod 查看 node 的資訊



4-1

開啟新的 Terminal

Ctrl + Alt + T





## 操作 Turtlesim

## 查看節點資訊

## 4. 利用 rosnod 查看 node 的資訊

指令：`$ rosnod list`

列出目前 ROS 系統啟動的 node

```
ros@practice:~$ rosnod list  
/rosout  
/turtlesim
```

4-2

輸入指令

4-3

目前啟動的 node



## 操作 Turtlesim

## 查看節點資訊

### 5.更改 ROS系統啟動中 node 的名稱

```
ros@practice:~$ rosrn turtlesim turtlesim_node  
[ INFO] [1520502221.562893958]: Starting turtles  
[ INFO] [1520502221.575128309]: Spawning turtle  
0.000000]  
^C  
ros@practice:~$
```

5-1

選擇啟動  
turtlesim\_node  
的 terminal

5-2

按下  +   
取消當前執行的指  
令



## 操作 Turtlesim

## 查看節點資訊

### 5.更改 ROS系統啟動中 node 的名稱

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

藍色字體：package 的名稱

綠色字體：node 名稱

紫色字體：特殊關鍵字，\_\_name 為 node 名稱

淺澄字體：remapping 參數語法

紅色字體：remapping 的參數

```
ros@practice:~$ rosrun turtlesim turtlesim_node __name:=my_turtle
[ INFO] [1520505217.336643511]: Starting turtlesim with node name
[ INFO] [1520505217.349923639]: Spawning turtle [turtle1] at x=[5.
0.000000]
```

5-3

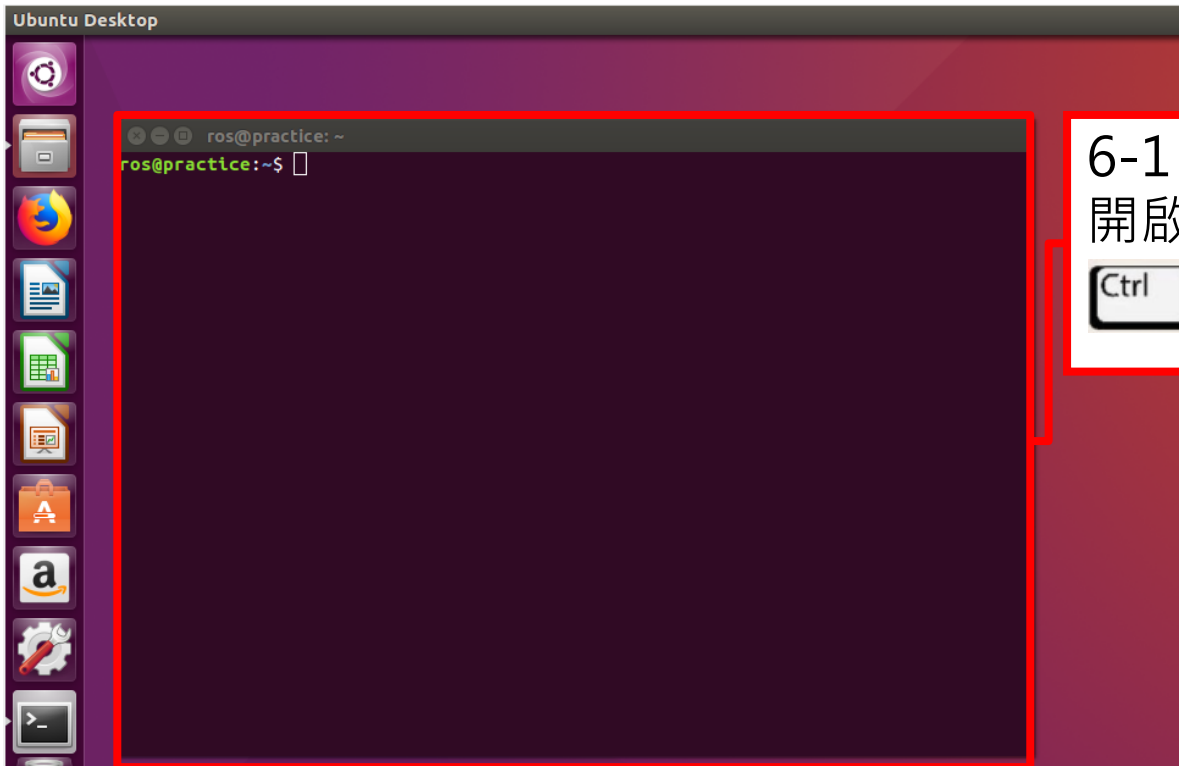
輸入指令



## 操作 Turtlesim

## 查看節點資訊

## 6. 利用 rosnode 查看 node 的資訊



6-1

開啟新的 Terminal

Ctrl + Alt + T



## 操作 Turtlesim

## 查看節點資訊

## 6. 利用 rosnod 查看 node 的資訊

指令：`$ rosnod list`

列出目前 ROS 系統啟動的 node

```
ros@practice:~$ rosnod list
```

```
/my_turtle  
/rosout  
/turtlesim
```

6-2  
輸入指令

6-3  
目前啟動的 node



## 操作 Turtlesim

## 查看節點資訊

## 6.利用 rosnod 查看 node 的資訊

```
ros@practice:~$ rosnod list  
/my_turtle  
/rosout  
/turtlesim
```

6-4

發現 node 「 turtlesim 」  
還存在，這是因為我們使用  
**Ctrl + C** 終止運行 node  
「 turtlesim 」，使ROS系統未  
更新到節點資訊





## 操作 Turtlesim

## 查看節點資訊

## 6.利用 rosnod 查看 node 的資訊

指令：`$ rosnod cleanup`

清除目前無法進行連線的 node  
(但 node 可能還是處於正常運作狀態)

```
ros@practice:~$ rosnod cleanup
ERROR: connection refused to [http://localhost:46043/]
Unable to contact the following nodes:
 * /turtlesim
Warning: these might include alive and functioning nodes, e.g.
rks.
Cleanup will purge all information about these nodes from the master.
Please type y or n to continue:
y
Unregistering /turtlesim
done
ros@practice:~$
```

6-5  
輸入指令

6-6  
輸入「y」  
確認執行



## 操作 Turtlesim

## 查看節點資訊

## 6. 利用 rosnod 查看 node 的資訊

指令：`$ rosnod list`

列出目前 ROS 系統啟動的 node

```
ros@practice:~$ rosnod list  
/my_turtle  
/rosout
```

6-7  
輸入指令

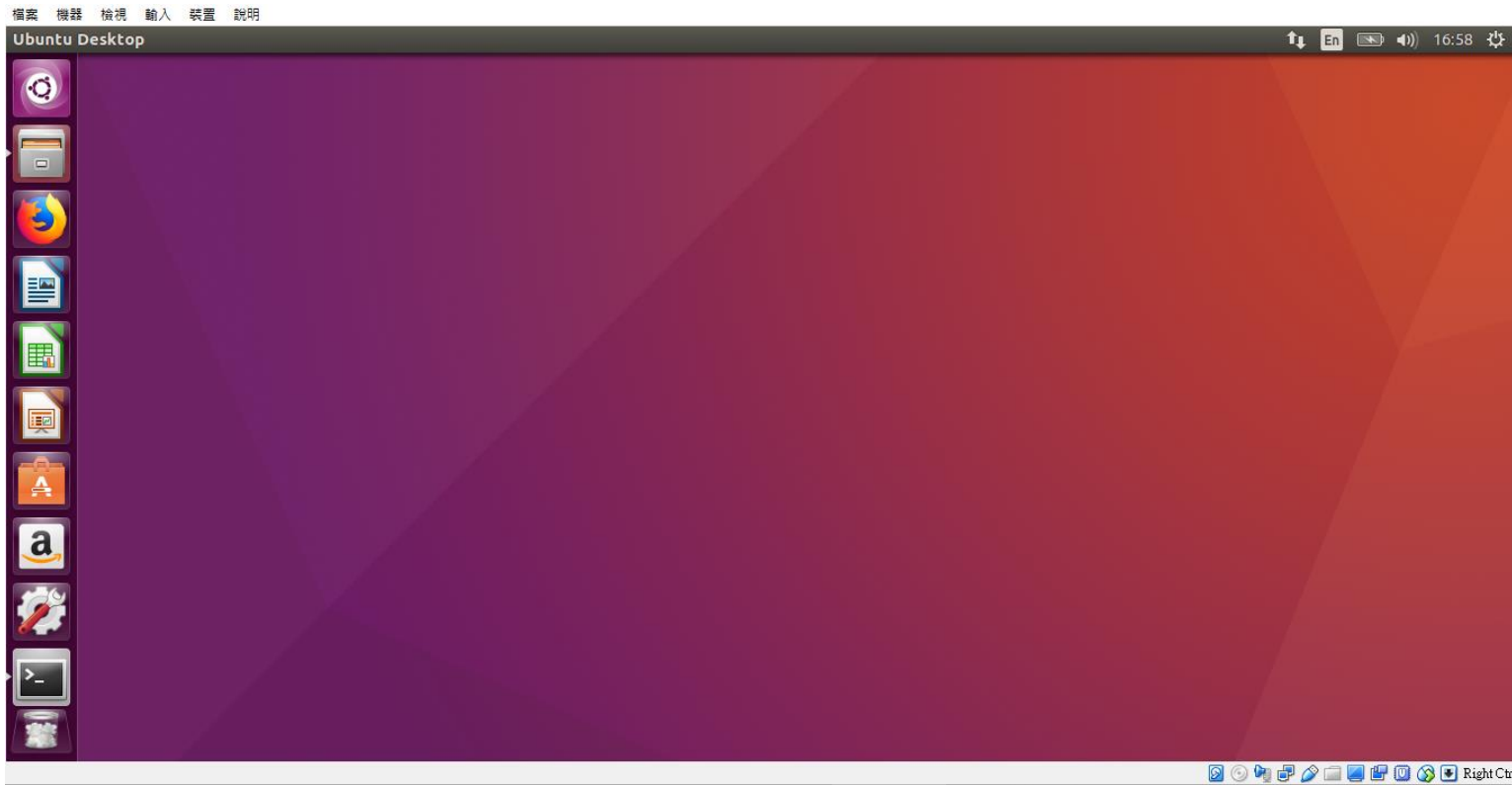
6-8  
發現只剩下可以連線到的 node



## 操作 Turtlesim

## 資料視覺化

# 1.關閉所有執行中的 Terminal



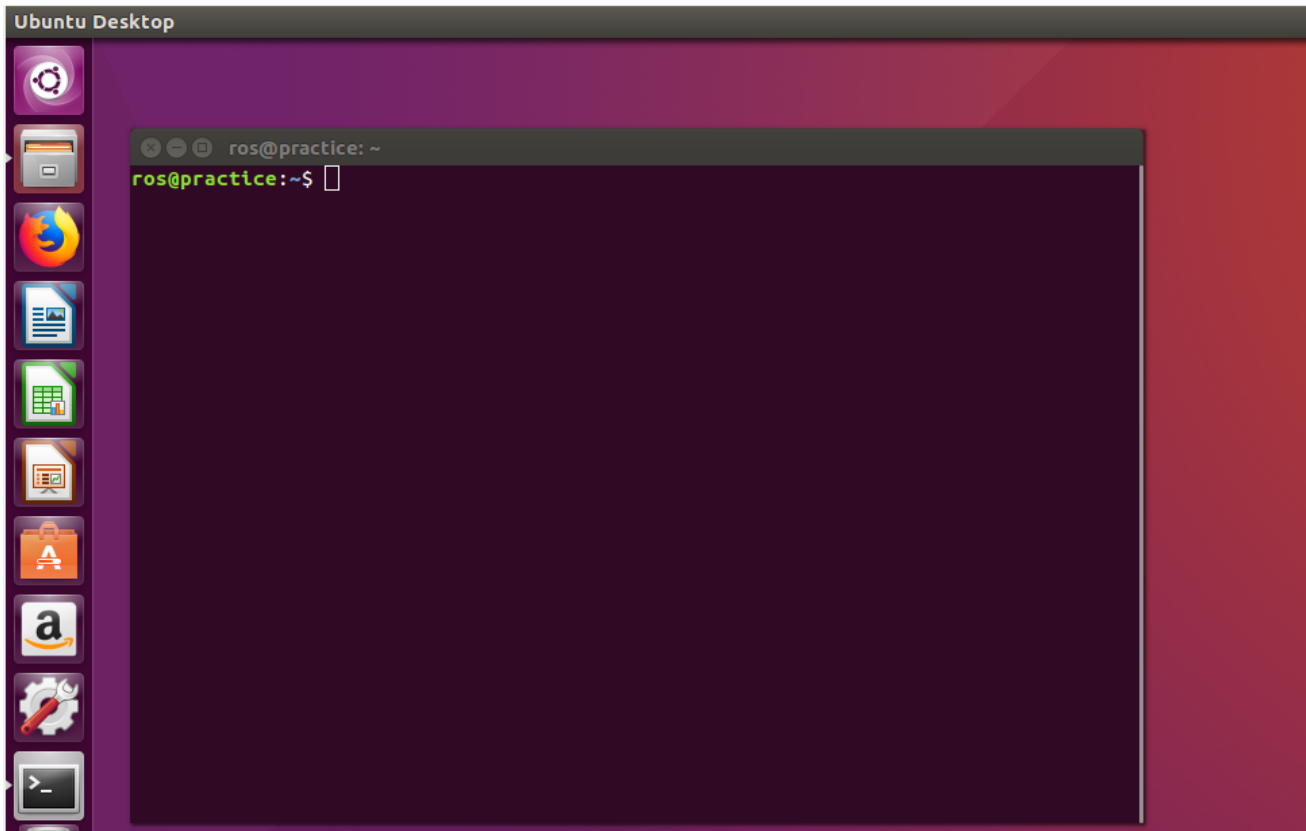
備註：可於第 3 張 PPT 查看 Service 的說明



## 操作 Turtlesim

## 資料視覺化

### 2. 開啟 Terminal + +





## 操作 Turtlesim

## 資料視覺化

## 3.開啟ROS系統的 Master Node

指令 : **\$ roscore**

```
roscore http://localhost:11311/  
ros@practice:~$ roscore  
... loading to /home/ros/.ros/log/d7f62824-22b0-11e8-9249-74da  
-practice-3711.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://localhost:42401/  
ros_comm version 1.12.12
```

3-1

輸入指令

3-2

ROS啟動  
Master  
Node





## 操作 Turtlesim

## 資料視覺化

4. 利用 rosrun 啟動位於



turtlesim

的



turtlesim\_node

指令：`$ rosrun turtlesim turtlesim_node`

藍色字體：package 的名稱

綠色字體：node 名稱

```
ros@practice:~$ rosrun turtlesim turtlesim_node
[ INFO] [1520502221.562893958]: Starting turtlesim with node name /turtlesim
[ INFO] [1520502221.575128309]: Spawning turtle [turtle1] at x=[5.544445], y=
0.000000]
```

4-1  
輸入指令





## 操作 Turtlesim

## 資料視覺化

### 4. 利用 rosrun 啟動位於



turtlesim

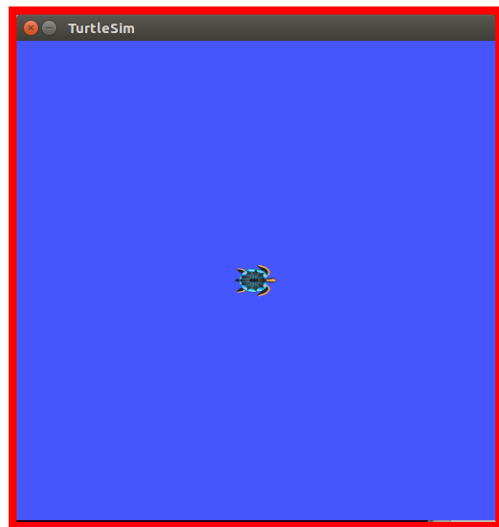
的



turtlesim\_node

```
ros@practice:~$ rosrun turtlesim turtlesim_node
```

```
[ INFO] [1520502221.562893958]: Starting turtlesim with node name /turtlesim  
[ INFO] [1520502221.575128309]: Spawning turtle [turtle1] at x=[5.544445], y=  
0.000000]
```



4-2



turtlesim\_node

啟動後資訊

4-3

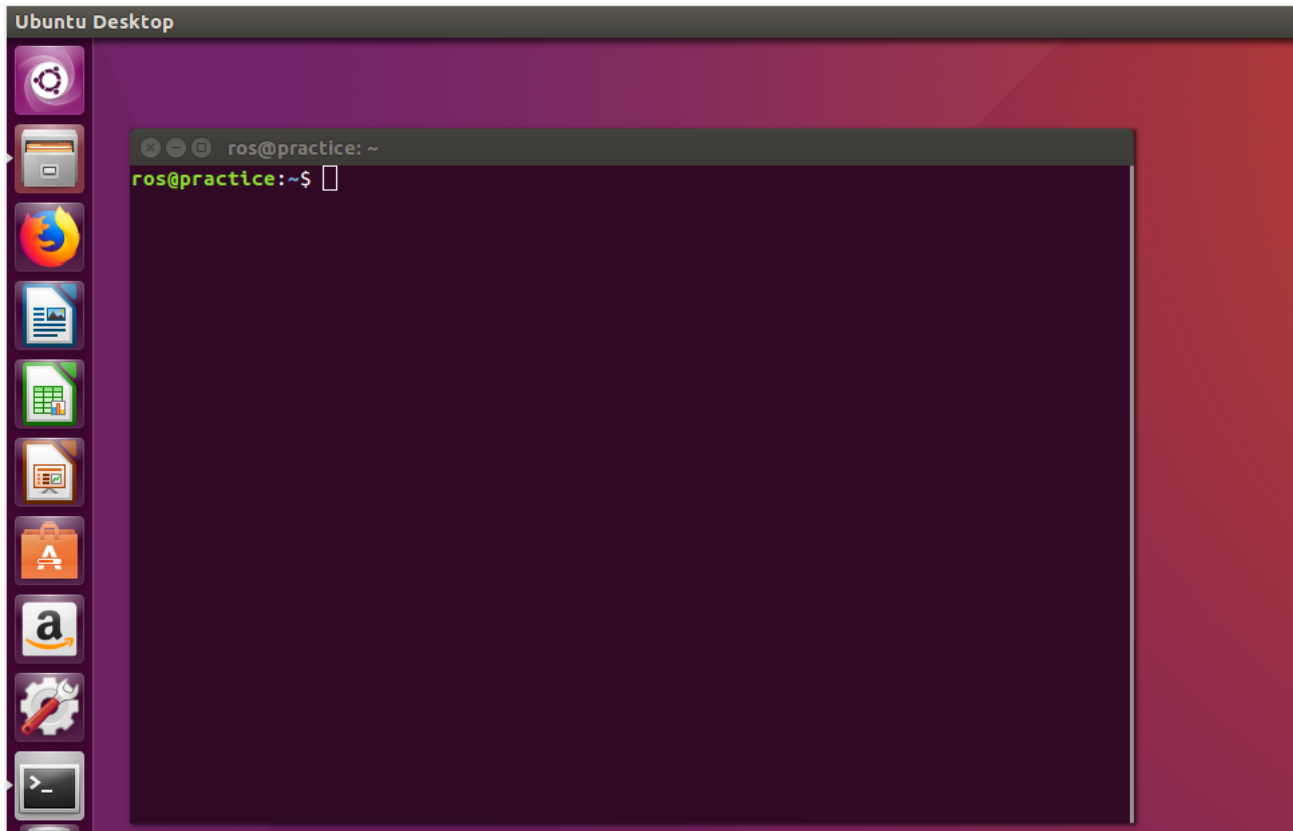
啟動後畫面  
(烏龜為隨機載入)



## 操作 Turtlesim

## 資料視覺化

### 5. 開啟新的 Terminal





## 操作 Turtlesim

## 資料視覺化

6. 利用 rosrund 啟動位於



turtlesim

的



turtle\_teleop\_key

指令：`$ rosrund turtlesim turtle_teleop_key`

藍色字體：`package` 的名稱

綠色字體：`node` 名稱

```
ros@practice:~$ rosrund turtlesim turtle_teleop_key
```

```
Reading from keyboard
```

```
-----  
Use arrow keys to move the turtle.
```



6-1

輸入指令

6-2

開始按下方向鍵  
按鈕移動烏龜



## 操作 Turtlesim

## 資料視覺化

6. 利用 `roslaunch` 啟動位於

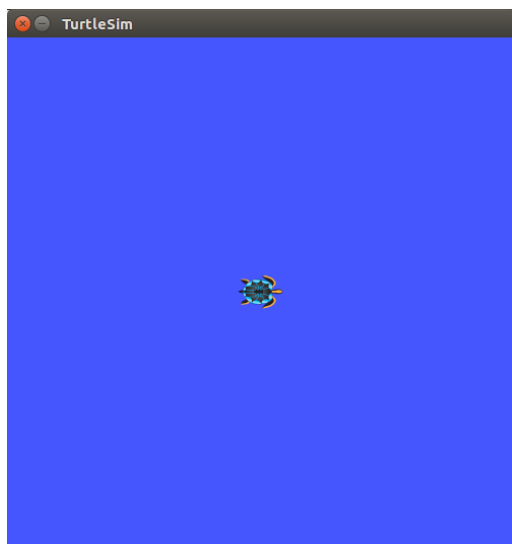


`turtlesim`

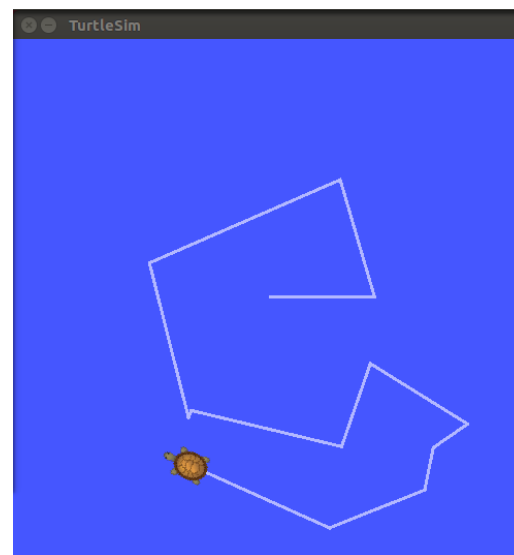
的



`turtle_teleop_key`



按下方向鍵後

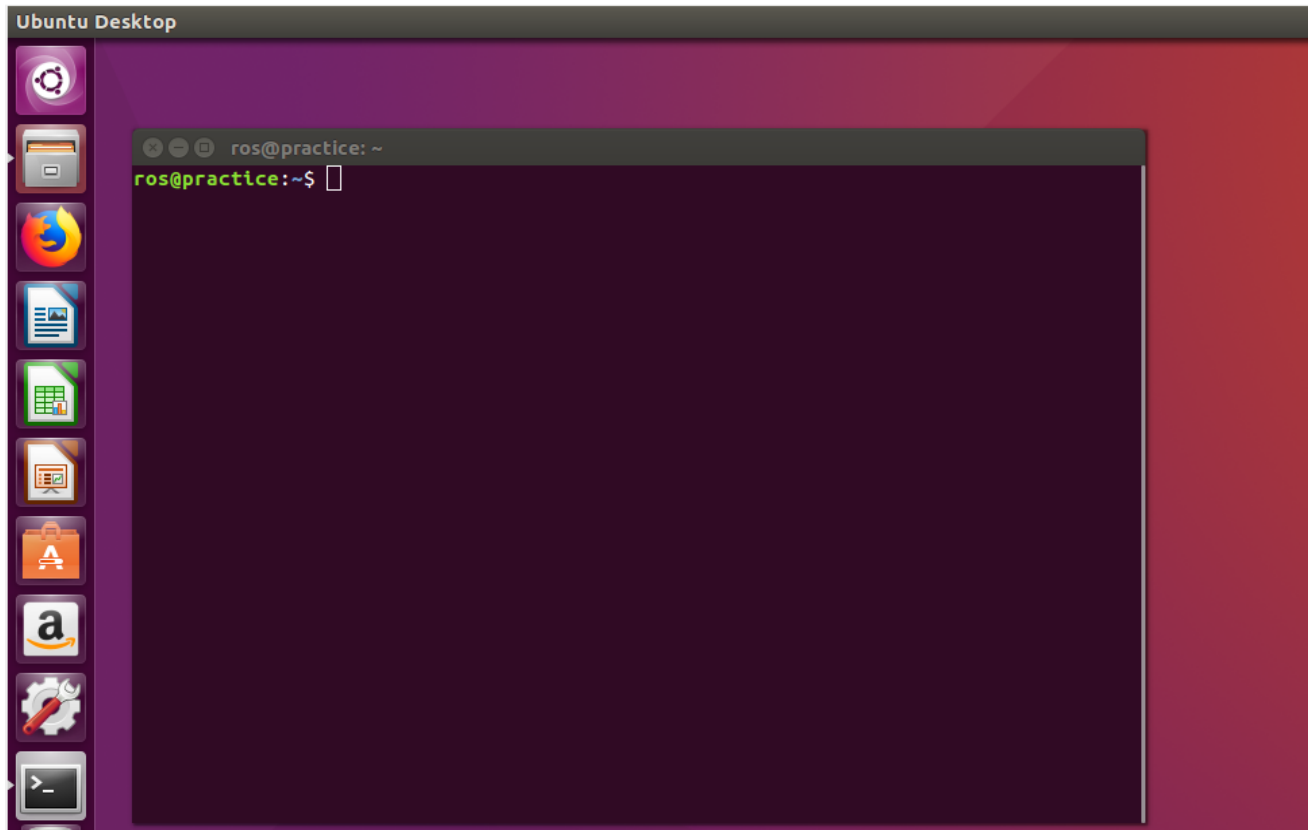
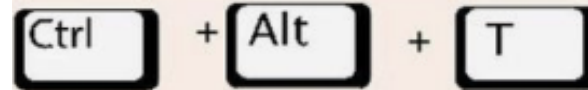




# 操作 Turtlesim

## 資料視覺化

### 7.開啟新的 Terminal





## 操作 Turtlesim

## 資料視覺化

8.利用 rosrund啟動位於



rqt\_graph

的



rqt\_graph

指令：`$ rosrund rqt_graph rqt_graph`

(藍色字體：package 的名稱、綠色字體：node 名稱)

利用



rqt\_graph

繪製ROS系統的動態流程圖(dynamic graph)

```
ros@practice:~$ rosrund rqt_graph rqt_graph
```

8-1

輸入指令





# 操作 Turtlesim

## 資料視覺化

8. 利用 rosrun 啟動位於



rqt\_graph

的



rqt\_graph

\* 將滑鼠移動到 /turtle1/command\_velocity

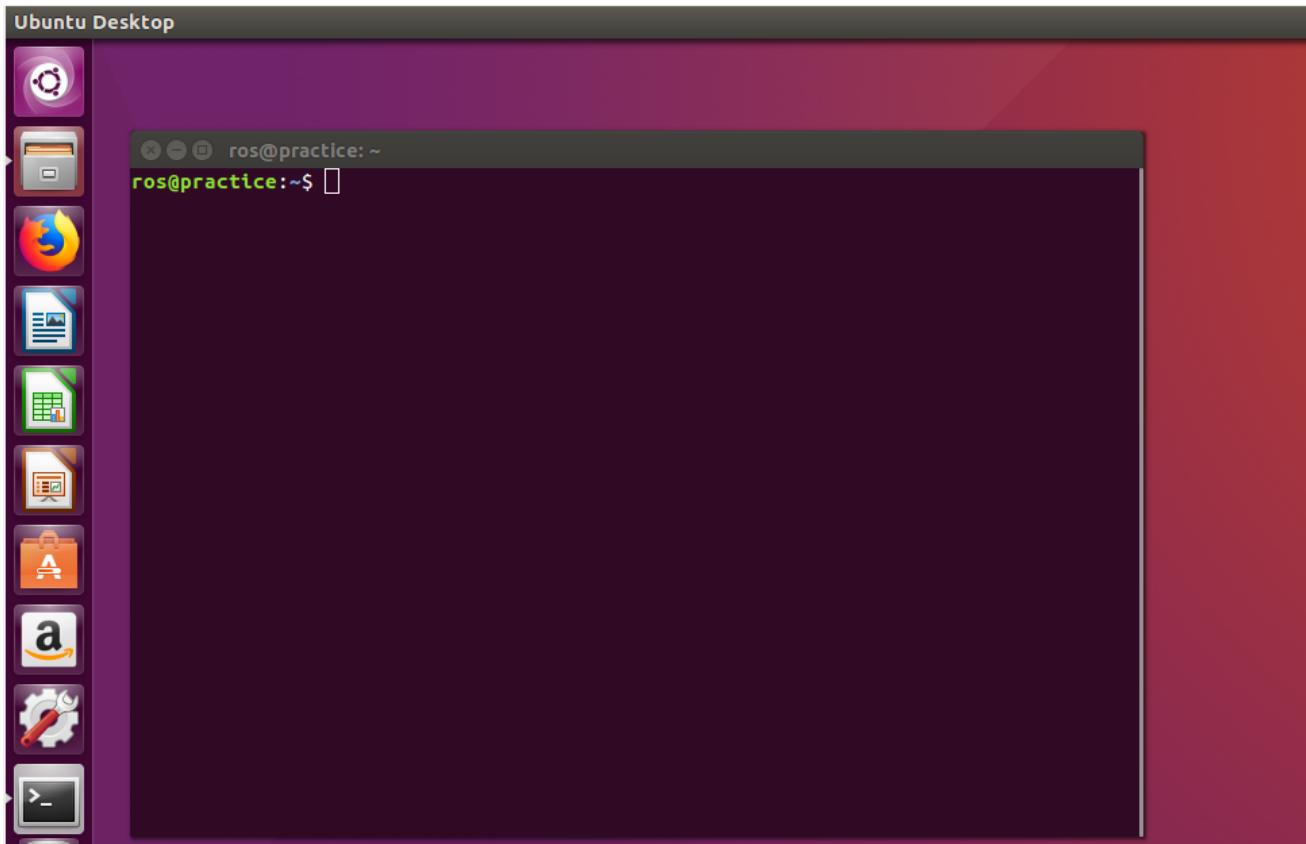




操作 Turtlesim

查看 Topic

## 9. 開啟新的 Terminal





操作 Turtlesim

查看 Topic

## 10. 利用 rostopic 查看 Topic 的資訊

指令：**\$ rostopic -h**

查看 rostopic 可以使用的功能

```
ros@practice:~$ rostopic -h
rostopic is a command-line tool for printing information about ROS Topics.
```

### commands:

rostopic bw	display bandwidth used by topic
rostopic delay	display delay of topic from timestamp in header
rostopic echo	print messages to screen
rostopic find	find topics by type
rostopic hz	display publishing rate of topic
rostopic info	print information about active topic
rostopic list	list active topics
rostopic pub	publish data to topic
rostopic type	print topic or field type

```
Type rostopic <command> -h for more detailed usage, e.g. 'rostopic echo -h'
```

10-1  
輸入指令

10-2  
功能說明



操作 Turtlesim

查看 Topic

## 10. 利用 rostopic 查看 Topic 的資訊

指令：`$ rostopic echo /turtle1/cmd_vel`

(紅色字體：Topic 的名稱)

顯示 `/turtle1/command_velocity` 被發布的資料

```
ros@practice:~$ rostopic echo /turtle1/cmd_vel
```

10-1  
輸入指令

無任何訊息出現是因為現在沒  
有任何資料被發布



操作 Turtlesim

查看 Topic

## 10. 利用 rostopic 查看 Topic 的資訊

移動到執行 `roslaunch turtlesim turtlesim_key` 的 Terminal 視窗，並按下任意方向鍵

```
ros@practice:~$ 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
---
```

10-2

回到 `$ rostopic echo /turtle1/cmd_vel`  
的 Terminal 視窗，即可查看到  
`/turtle1/cmd_vel` 發布的資料

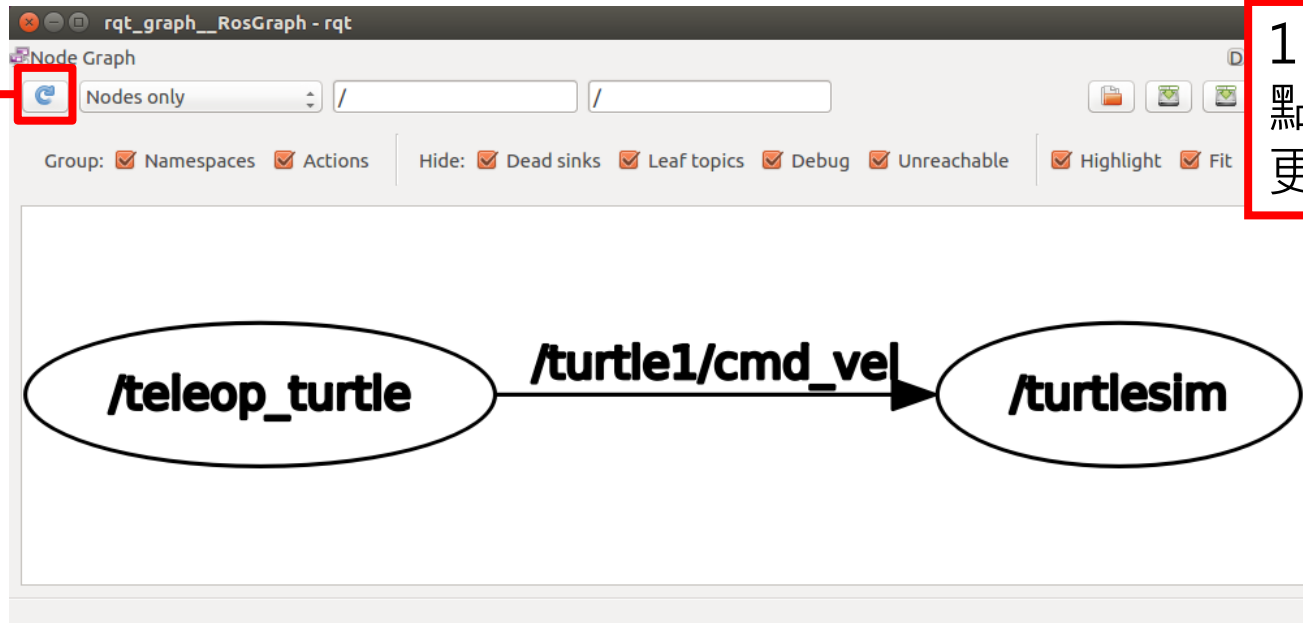




操作 Turtlesim

查看 Topic

## 11. 查看ROS系統動態流程圖(dynamic graph)

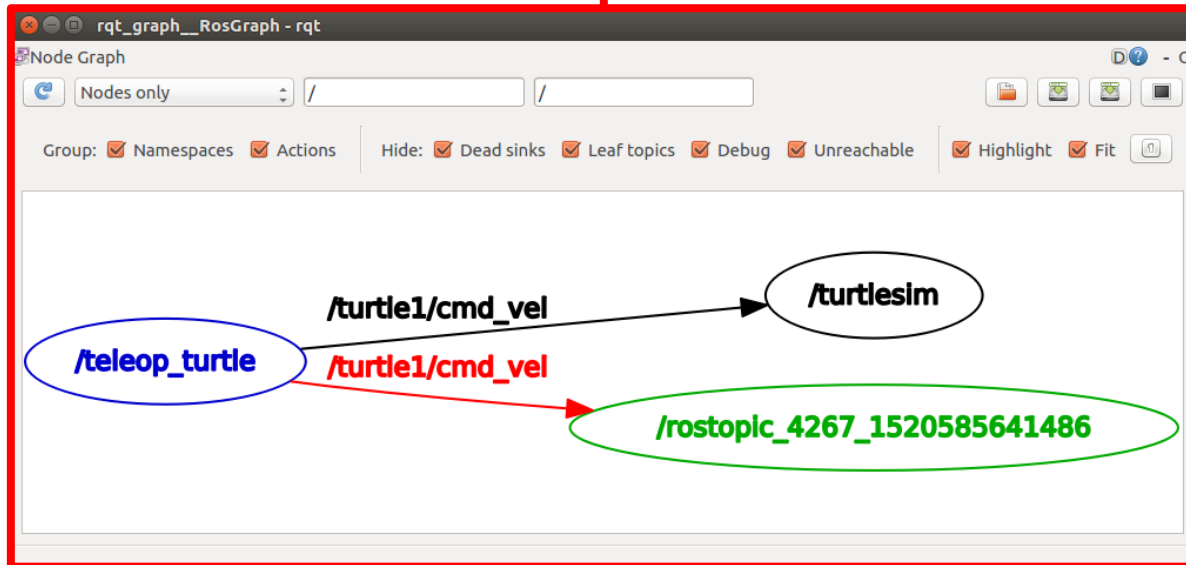
11-1  
點擊  
更新按鈕



操作 Turtlesim

查看 Topic

## 11.查看ROS系統動態流程圖(dynamic graph)



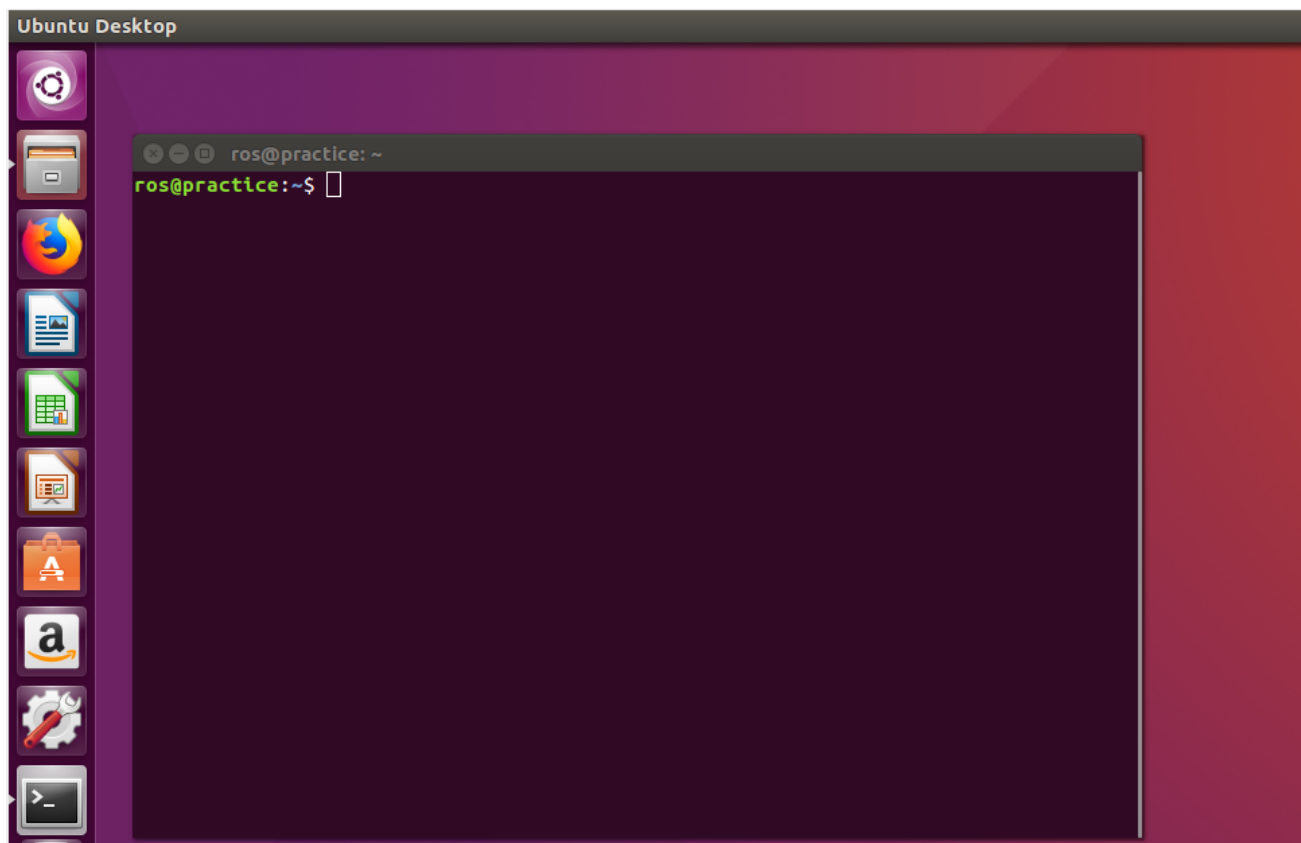
11-1  
動態流程  
圖更新後  
畫面



操作 Turtlesim

查看 Topic

## 12.開啟新的 Terminal





操作 Turtlesim

查看 Topic

## 13.使用 rostopic查看 Topic 的資訊

指令：**\$ rostopic list -h**

查看 rostopic list 的指令說明

```
ros@practice:~$ rostopic list -h
Usage: rostopic list [/namespace]
```

Options:

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

13-1  
輸入指令

13-2  
功能說明



操作 Turtlesim

查看 Topic

## 13.使用 rostopic查看 Topic 的資訊

指令：**\$ rostopic list -v**

查看 Publisher topic 與 Subscriber topic

```
ros@practice:~$ rostopic list -v
```

Published topics:

```
* /turtle1/color_sensor [turtlesim/Color] 1 publisher  
* /turtle1/cmd_vel [geometry_msgs/Twist] 1 publisher  
* /rosout [roscpp_msgs/Log] 3 publishers  
* /rosout_agg [roscpp_msgs/Log] 1 publisher  
* /turtle1/pose [turtlesim/Pose] 1 publisher
```

Subscribed topics:

```
* /turtle1/cmd_vel [geometry_msgs/Twist] 2 subscribers  
* /rosout [roscpp_msgs/Log] 1 subscriber
```

13-3  
輸入指令

13-4  
Topic的資訊





操作 Turtlesim

查看 Topic

## 13. 使用 rostopic 查看 Topic 的資訊

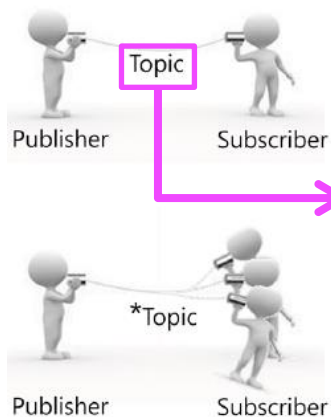
指令：`$ rostopic type /turtle1/cmd_vel`

查看 Messages 的資訊

```
ros@practice:~$ rostopic type /turtle1/cmd_vel  
geometry_msgs/Twist
```

13-3  
輸入指令

13-4  
Topic 的資訊



Topic 的  
資料型態依照  
Messages 決定





操作 Turtlesim

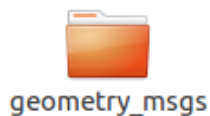
查看 Topic

## 14.使用 rosmmsg 查看 message 的資訊

指令：`$ rosmmsg show geometry_msgs/Twist`

紫色字體：Message 的名稱

顯示位於



勾



Twist.msg

```
ros@practice:~$ 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
```

14-1  
輸入指令14-2  
Message的資訊



操作 Turtlesim

查看 Topic

## 15.使用 rostopic 發布 Topic

指令：

```
$ rostopic pub -1 /turtle1/cmd_vel geometry_msgs/Twist -- '[2.0, 0.0, 0.0]' '[0.0, 0.0, 1.8]'
```

綠色字體：Topic 發布的次數

紅色字體：Topic 的名稱

紫色字體：Message 的名稱

淺澄字體：告知解析器之後的選項為參數

藍色字體：輸入的參數

```
ros@practice:~$ rostopic pub -1 /turtle1/cmd_vel geometry_msgs/Twist  
-- '[2.0, 0.0, 0.0]' '[0.0, 0.0, 1.8]'  
publishing and latching message for 3.0 seconds
```

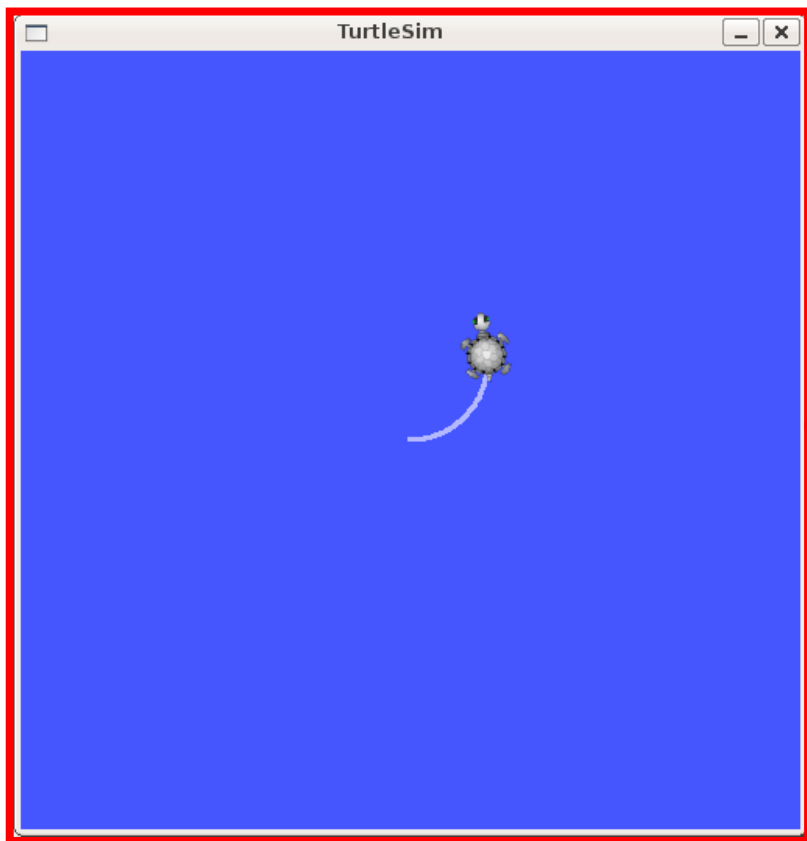
15-1  
輸入指令



操作 Turtlesim

查看 Topic

## 15.使用 rostopic 發布 Topic



15-2

烏龜開始按照發布的 Topic 開始移動



操作 Turtlesim

查看 Topic

## 15.使用 rostopic 發布 Topic

指令：`$ rostopic pub -h`

查看 rostopic pub 的功能指令

```
ros@practice:~$ rostopic pub -h
Usage: rostopic pub /topic type [args...]

Options:
  -h, --help            show this help message and exit
  -v                    print verbose output
  -r RATE, --rate=RATE  publishing rate (hz). For -f and stdin input, this
                        defaults to 10. Otherwise it is not set.
  -1, --once            publish one message and exit
  -f FILE, --file=FILE  read args from YAML file (Bagy)
  -l, --latch           enable latching for -f, -r and piped input. It
                        latches the first message.
  -s, --substitute-keywords
                        When publishing with a rate, performs keywords (now
                        or 'auto') substitution for each message
```

15-3

輸入指令

15-4

rostopic pub  
詳細功能說明



操作 Turtlesim

查看 Topic

## 15.使用 rostopic 發布 Topic

指令：

```
$ rostopic pub /turtle1/cmd_vel geometry_msgs/Twist -r 1 -- '[2.0, 0.0, 0.0]' '[0.0, 0.0, -1.8]'
```

綠色字體：Topic 發布的次數

紅色字體：Topic 的名稱

紫色字體：Message 的名稱

淺澄字體：告知解析器之後的選項為參數

藍色字體：輸入的參數

```
ros@practice:~$ rostopic pub -1 /turtle1/cmd_vel geometry_msgs/Twist  
-- '[2.0, 0.0, 0.0]' '[0.0, 0.0, 1.8]'  
publishing and latching message for 3.0 seconds
```

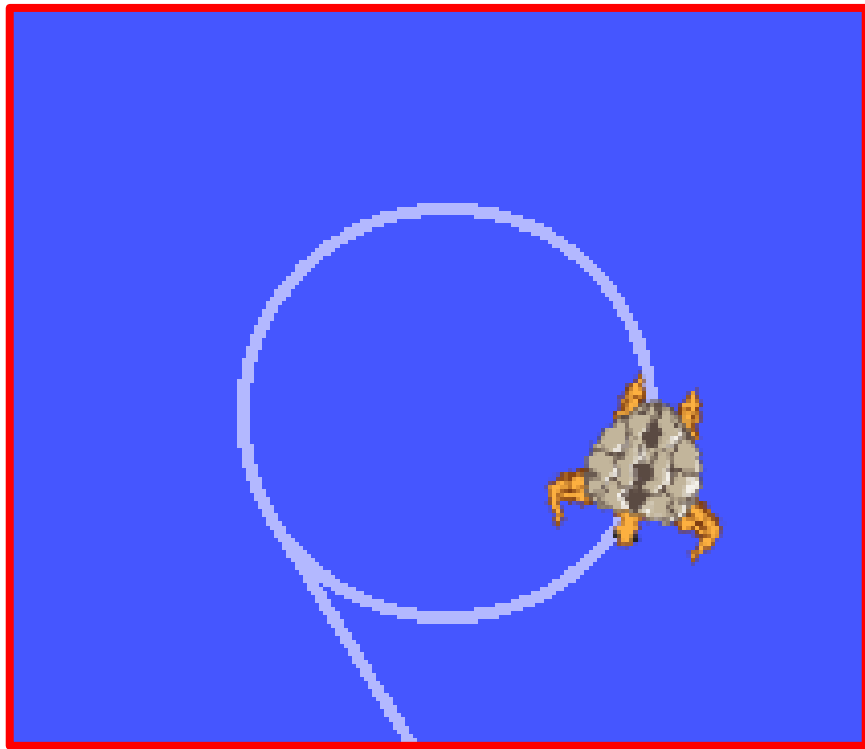
15-5  
輸入指令



操作 Turtlesim

查看 Topic

## 15.使用 rostopic 發布 Topic



15-6

烏龜開始按照發布的 Topic 開始移動



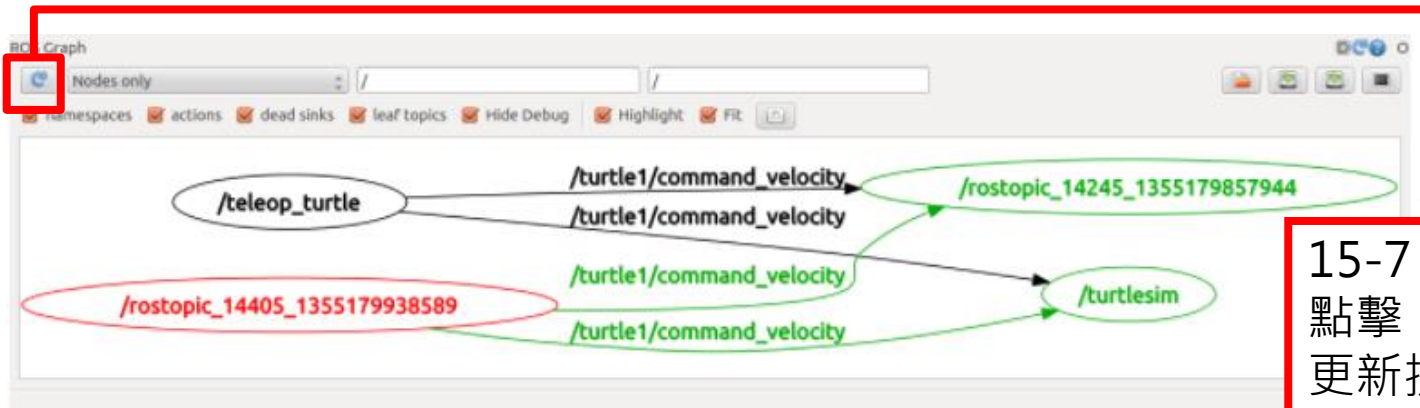


操作 Turtlesim

查看 Topic

## 15.使用 rostopic 發布 Topic

移動到 rqt\_graph 所繪製的  
動態流程圖(dynamic graph)



15-7  
點擊  
更新按鈕



操作 Turtlesim

查看 Topic

## 15.使用 rostopic 發布 Topic

指令：`$ rostopic hz /turtle1/pose`

紅色字體：Topic 的名稱

查看 Topic 的發布頻率

```
ros@practice:~$ rostopic hz /turtle1/pose
subscribed to [/turtle1/pose]
average rate: 62.542
  min: 0.014s max: 0.017s std dev: 0.00062s window: 59
average rate: 62.534
  min: 0.014s max: 0.018s std dev: 0.00069s window: 122
average rate: 62.510
  min: 0.013s max: 0.019s std dev: 0.00076s window: 185
average rate: 62.507
  min: 0.012s max: 0.020s std dev: 0.00084s window: 247
average rate: 62.508
  min: 0.012s max: 0.020s std dev: 0.00079s window: 310
average rate: 62.503
  min: 0.012s max: 0.020s std dev: 0.00077s window: 372
average rate: 62.501
  min: 0.012s max: 0.020s std dev: 0.00075s window: 435
```

15-8  
輸入指令

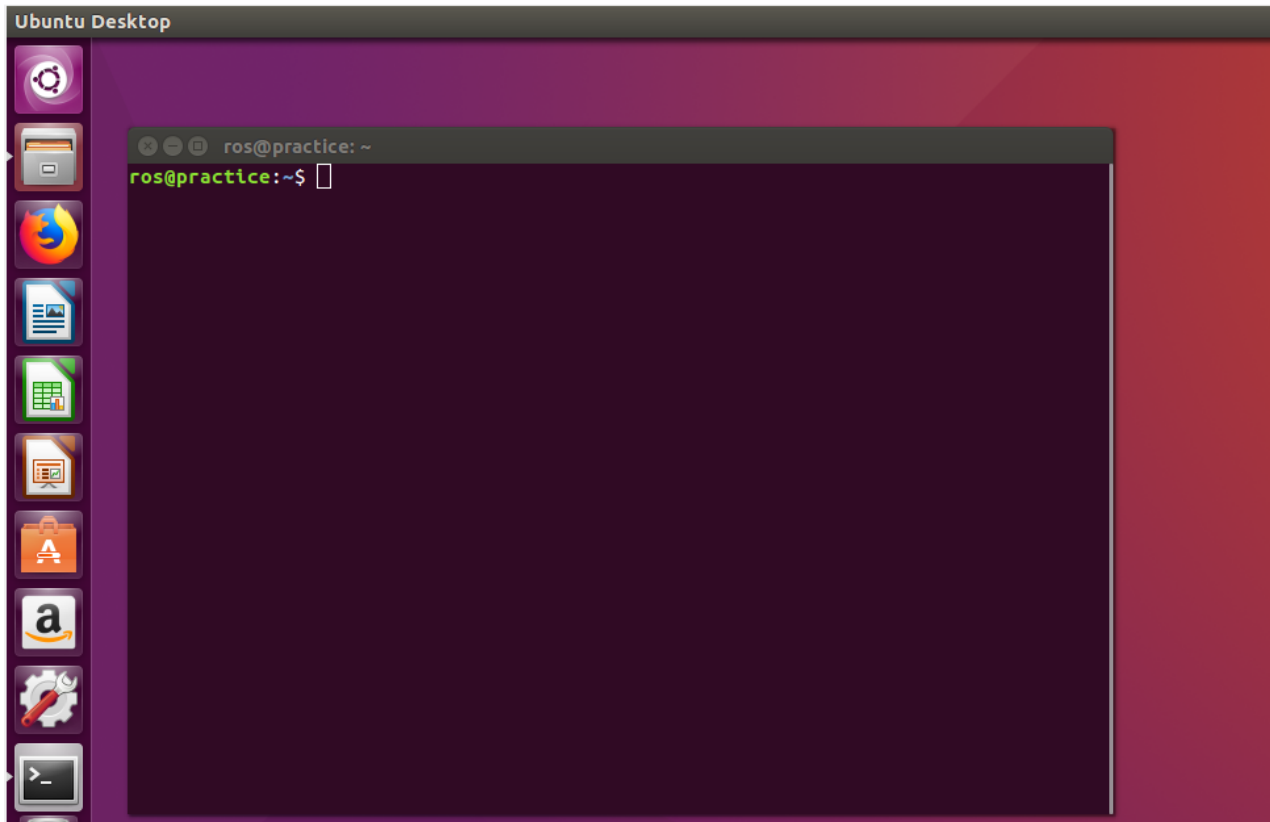
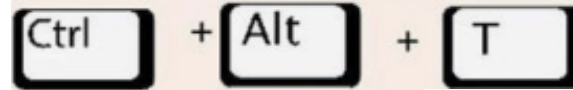
15-9  
Topic  
發布的頻率  
資訊



操作 Turtlesim

查看 Topic

## 16.開啟新的 Terminal





操作 Turtlesim

查看 Topic

## 17.使用 rqt\_plot 觀測 scrolling time plot

指令：`$ rostopic echo /turtle1/pose`

紅色字體：Topic 的 名稱

利用



rqt\_plot

繪製 Topic 發布資料的 scrolling time plot

```
ros@practice:~$ rostopic echo /turtle1/pose
```

```
x: 5.80983304977
```

```
y: 3.5617544651
```

```
theta: -5.39927577972
```

```
linear_velocity: 2.0
```

```
angular_velocity: -1.79999995232
```

17-1

輸入指令

17-2

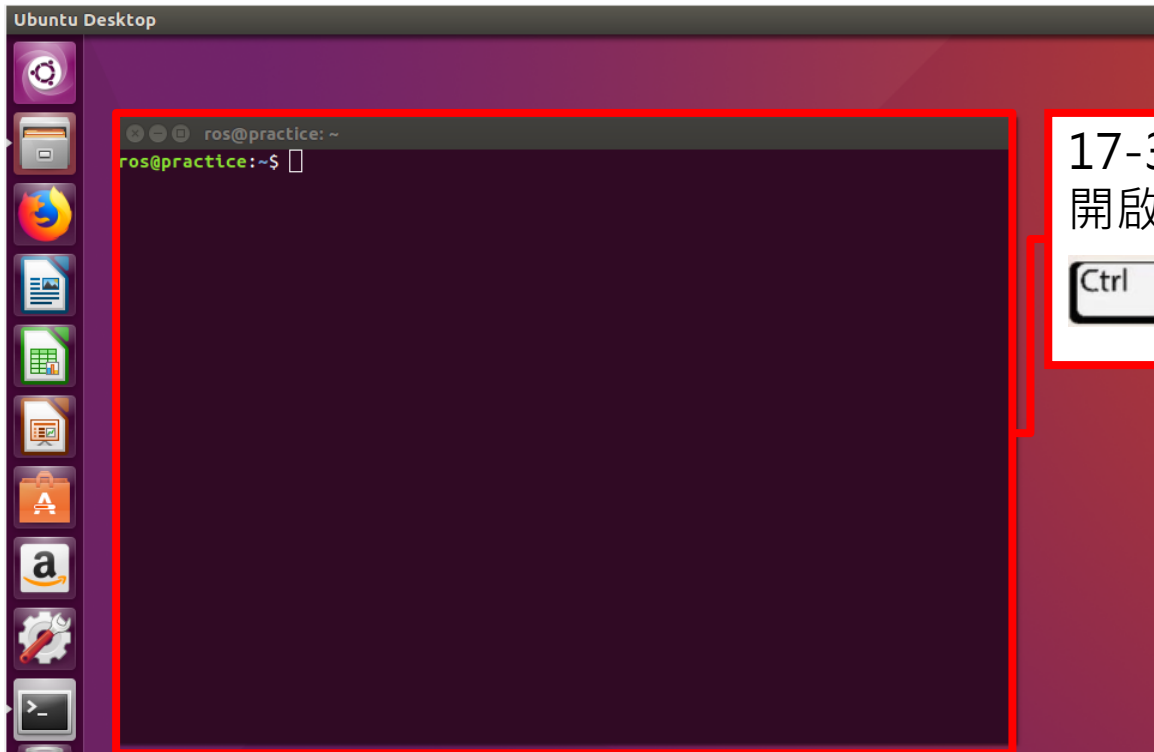
Topic 的  
資料顯示



操作 Turtlesim

查看 Topic

## 17.使用 rqt\_plot 觀測 scrolling time plot



17-3

開啟新的 Terminal

Ctrl + Alt + T





操作 Turtlesim

查看 Topic

## 17.使用 rqt\_plot 觀測 scrolling time plot

指令：`$ rosrun rqt_plot rqt_plot`

藍色字體：package 的名稱、綠色字體：node 名稱

利用



rqt\_plot

繪製 Topic 發布資料的 scrolling time plot

```
ros@practice:~$ rosrun rqt_plot rqt_plot
```

17-4  
輸入指令

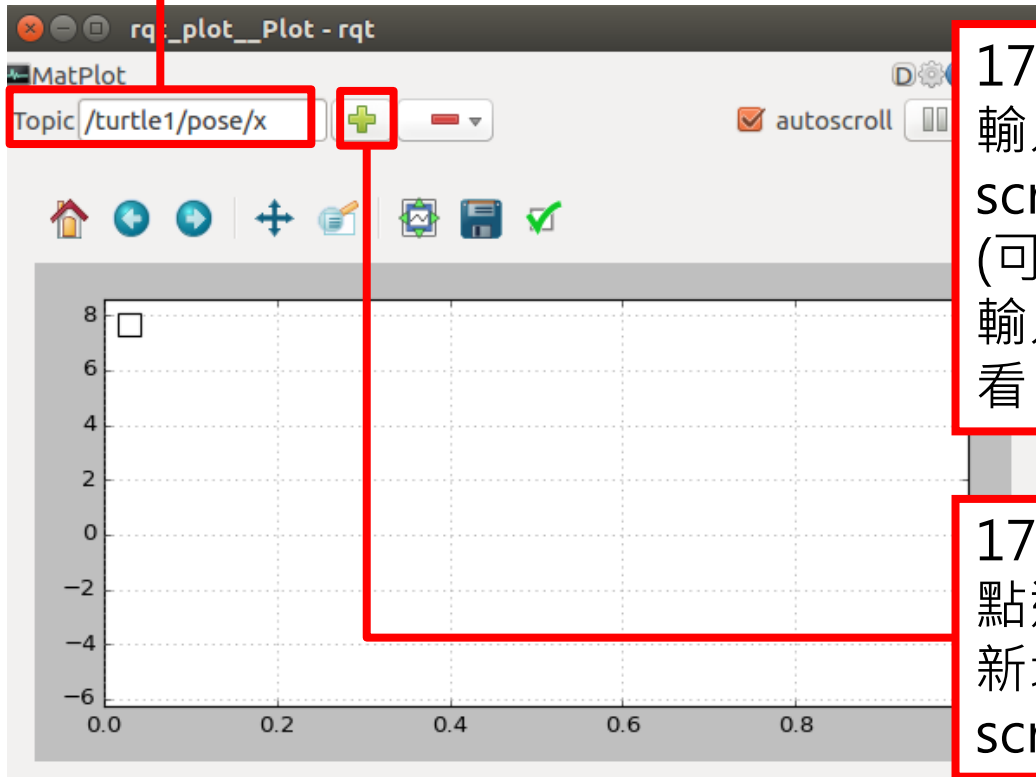




操作 Turtlesim

查看 Topic

## 17.使用 rqt\_plot 觀測 scrolling time plot



17-5

輸入 Topic 查看  
scrolling time plot  
(可以開啟新終端機，  
輸入 rostopic list 查  
看目前 Topic)

17-6

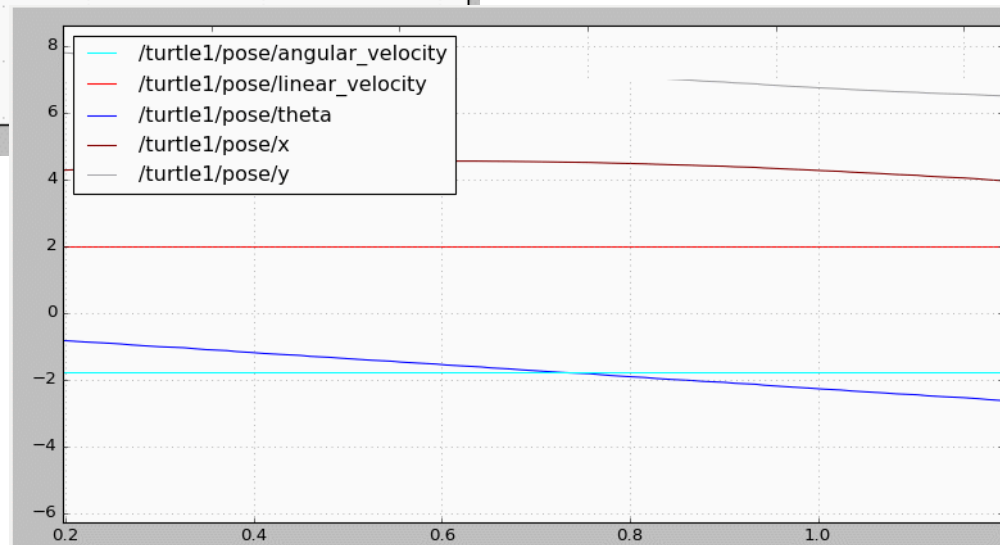
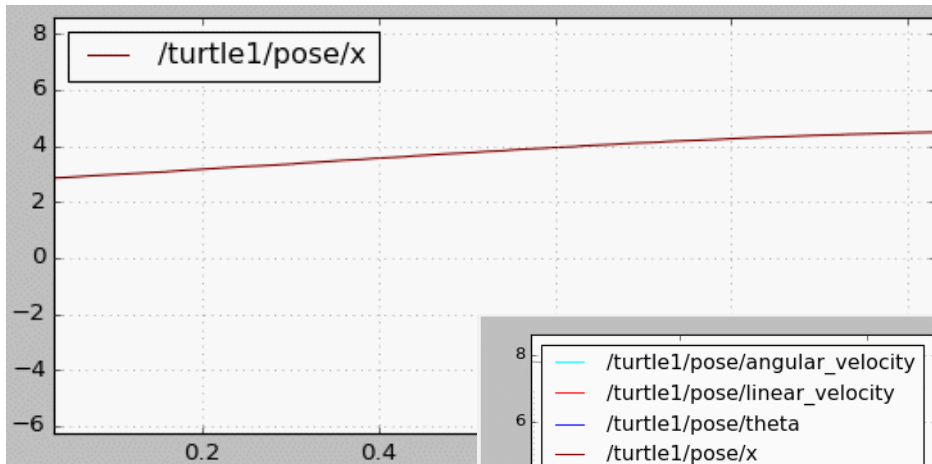
點選「+」圖示，  
新增該 Topic 的  
scrolling time plot



操作 Turtlesim

查看 Topic

## 17. 使用 rqt\_plot 觀測 scrolling time plot





## 操作 Turtlesim

## 執行 Services

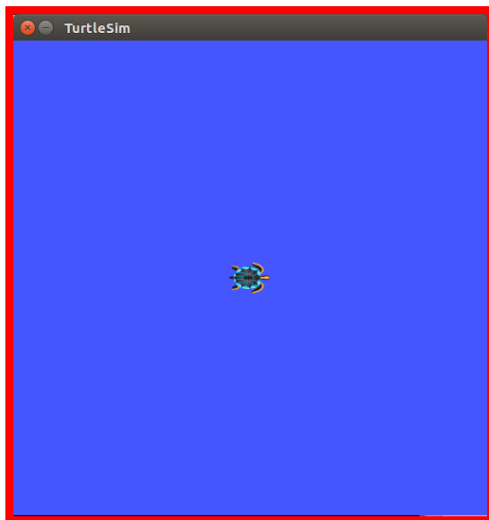
## 1. 確認



turtlesim\_node

依然被執行

```
ros@practice:~$ roscpp turtlesim turtlesim_node  
[ INFO] [1520502221.562893958]: Starting turtlesim with node name /turtlesim  
[ INFO] [1520502221.575128309]: Spawning turtle [turtle1] at x=[5.544445], y=  
0.000000]
```



1-1

移動到執行



turtlesim\_node

的 Terminal

1-2

執行的畫面

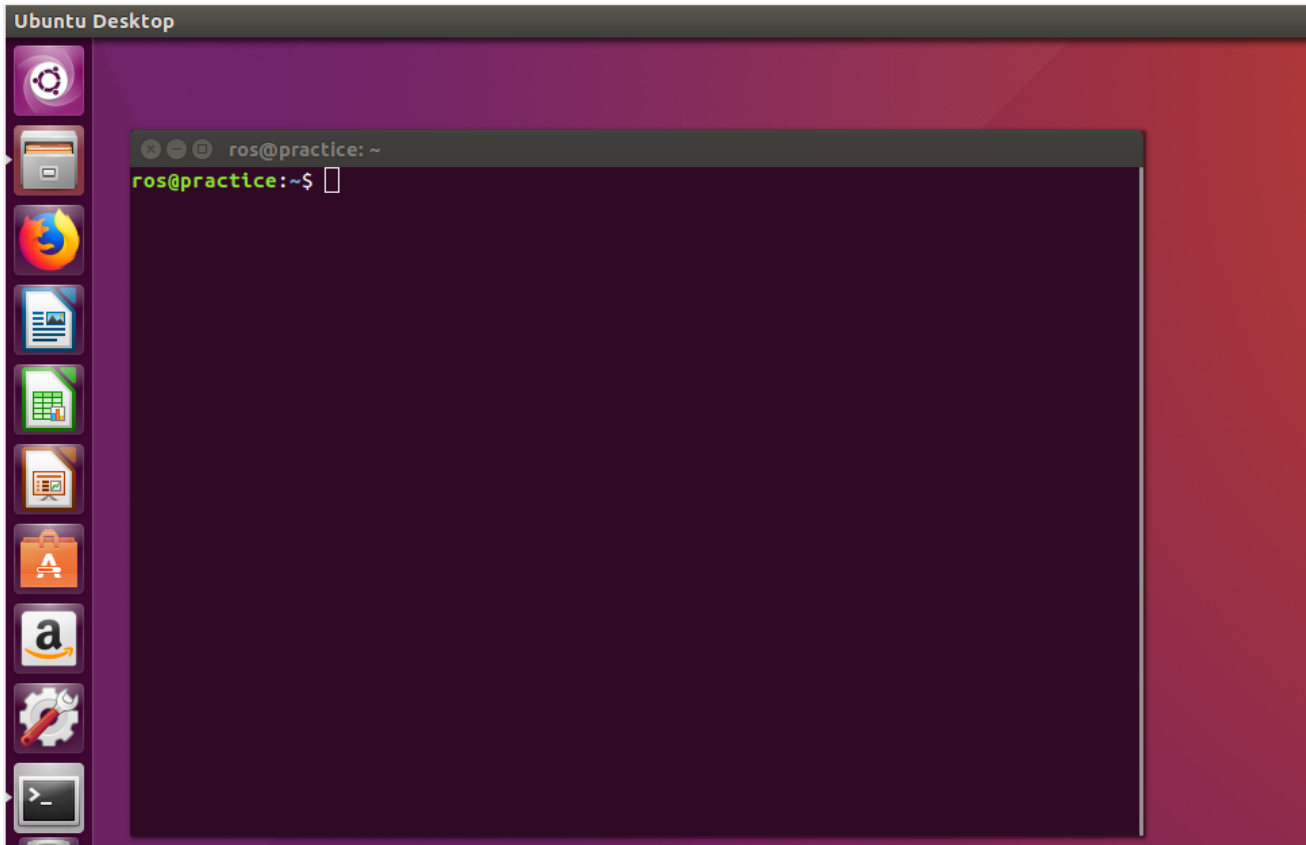
備註：可於第 4 張 PPT 查看 Service 的說明



## 操作 Turtlesim

## 執行 Services

### 2. 開啟新的 Terminal





## 操作 Turtlesim

## 執行 Services

## 3.使用 rosservice

指令：`$ rosservice -h`

查看 rosservice 可以使用的工具

```
ros@practice:~$ rosservice -h
```

```
Commands:
```

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

```
Type rosservice <command> -h for more detailed usage, e.g. 'rosservice call -h'
```

3-1

輸入指令

3-2

資料顯示





## 操作 Turtlesim

## 執行 Services

## 3.使用 rosservice

指令：**\$ rosservice list**

查看目前可以使用的 service

```
ros@practice:~$ rosservice list
```

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

3-3  
輸入指令

3-4  
資料顯示





## 操作 Turtlesim

## 執行 Services

## 3. 使用 rosservice

指令：`$ rosservice type /spawn`

棕色字體：可使用的 service

查看 service 所使用的資料型態依賴

```
ros@practice:~$ rosservice type /spawn  
turtlesim/Spawn
```

可利用 `rossrv show` 查看依賴的內容

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

3-5  
輸入指令

3-6  
資料顯示



## 操作 Turtlesim

## 查看ROS日誌紀錄

## 3.使用 rosservice

指令：`$ rosservice call /spawn 2 2 0.2`

棕色字體：可使用的 service

藍色字體：輸入的參數

使用 service 提供的服務 /spawn

```
ros@practice:~$ rosservice call /spawn 2 2 0.2 "  
name: "turtle2"
```



在 Turtlesim  
視窗產生  
第二隻烏龜  
名字為  
turtle2

3-7  
輸入指令

3-8  
Service的回應



操作 Turtlesim

查看ROS日誌紀錄

# 1.關閉所有執行中的 Terminal

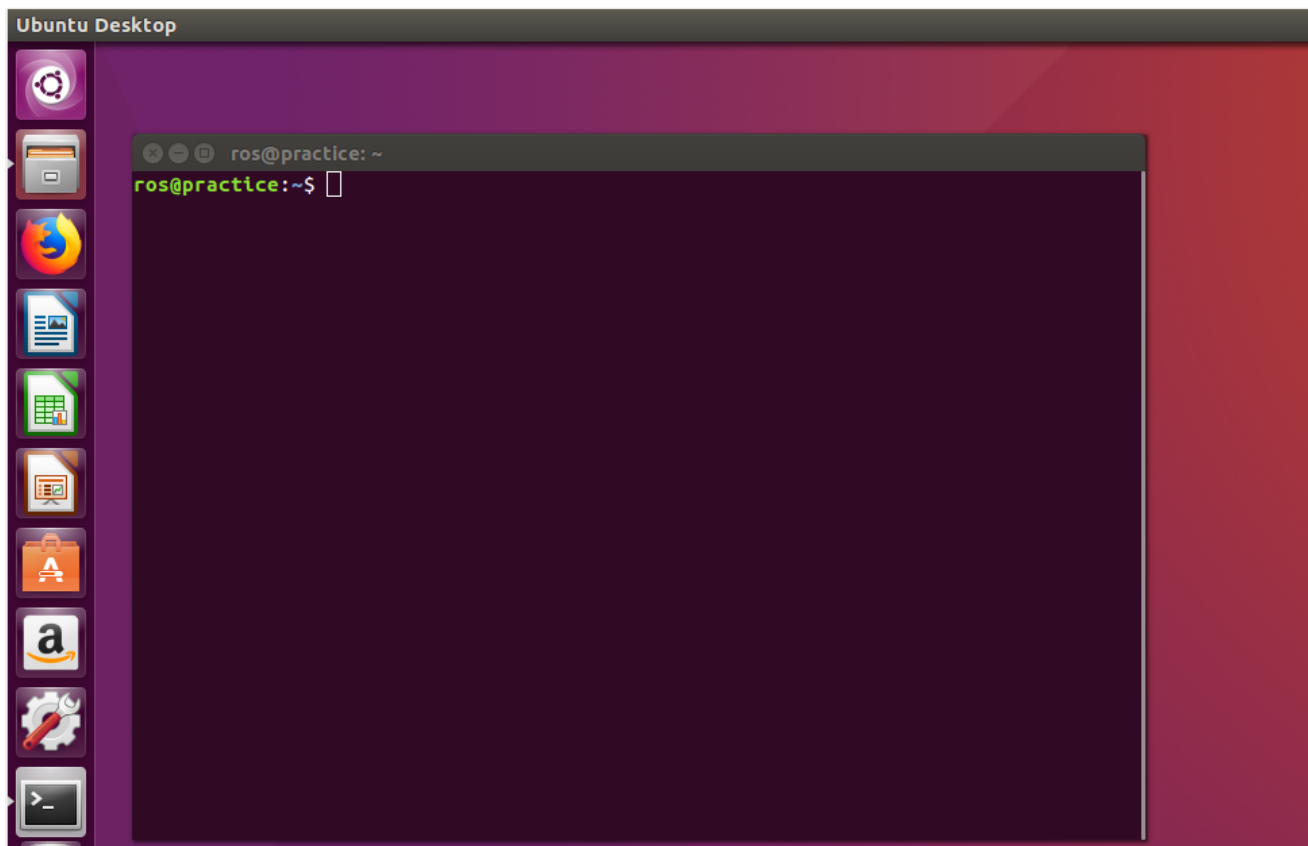




## 操作 Turtlesim

## 查看ROS日誌紀錄

### 2.開啟新的 Terminal





## 操作 Turtlesim

## 查看ROS日誌紀錄

## 3.開啟ROS系統的 Master Node

指令：**\$ roscore**

```
roscore http://localhost:11311/  
ros@practice:~$ roscore  
... loading to /home/ros/.ros/loa/d7f62824-22b0-11e8-9249-74da  
-practice-3711.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://localhost:42401/  
ros_comm version 1.12.12
```

3-1  
輸入指令

3-2  
ROS啟動  
Master  
Node

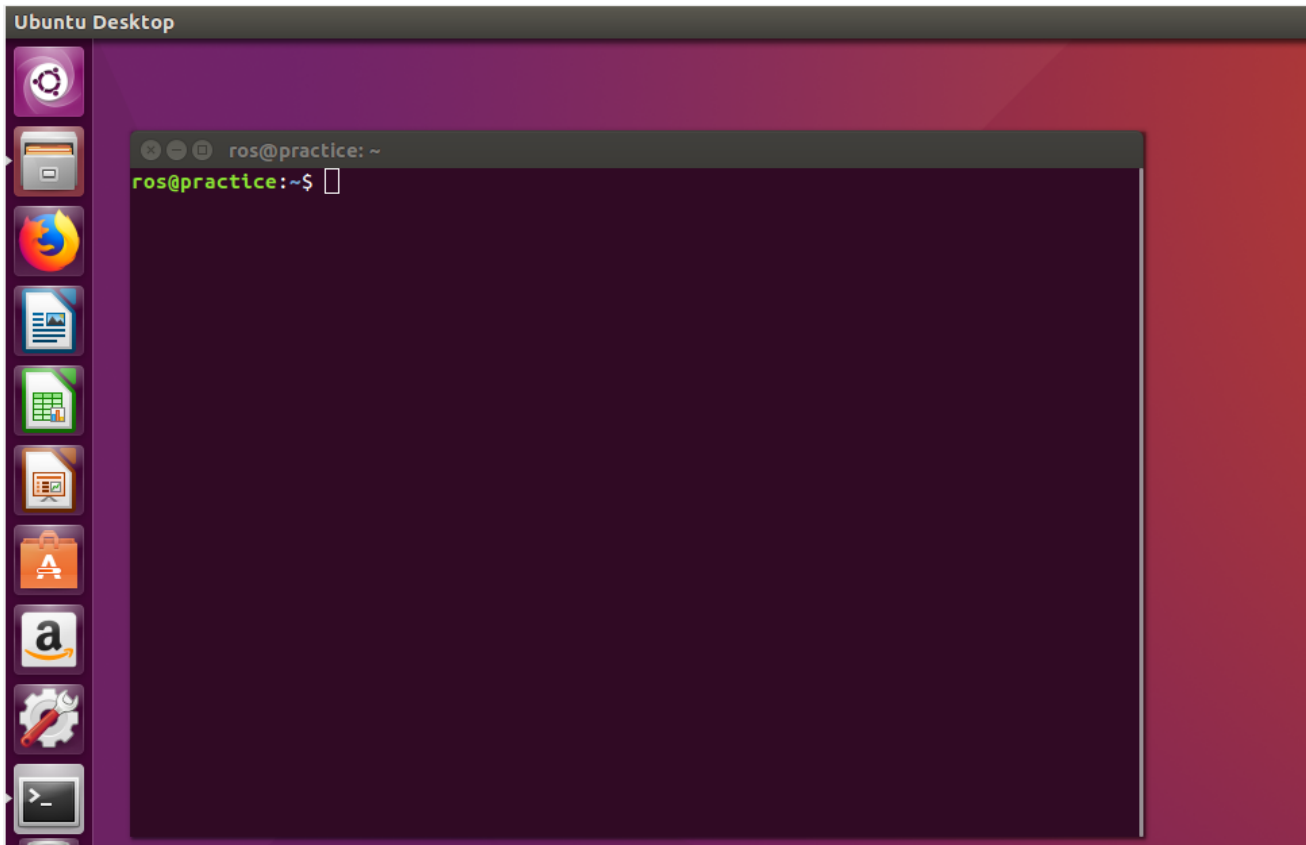




## 操作 Turtlesim

## 查看ROS日誌紀錄

### 4.開啟新的 Terminal







## 操作 Turtlesim

## 查看ROS日誌紀錄

5.利用 rosrund啟動位於



rqt\_console

的



rqt\_console

指令：`$ rosrund rqt_console rqt_console`

(藍色字體：package 的名稱、綠色字體：node 名稱)

利用



rqt\_console

查看 node 的日誌 (logging) 訊息

```
ros@practice:~$ rosrund rqt_console rqt_console
```

5-1

輸入指令



## 操作 Turtlesim

## 查看ROS日誌紀錄

5.利用 rosrun啟動位於

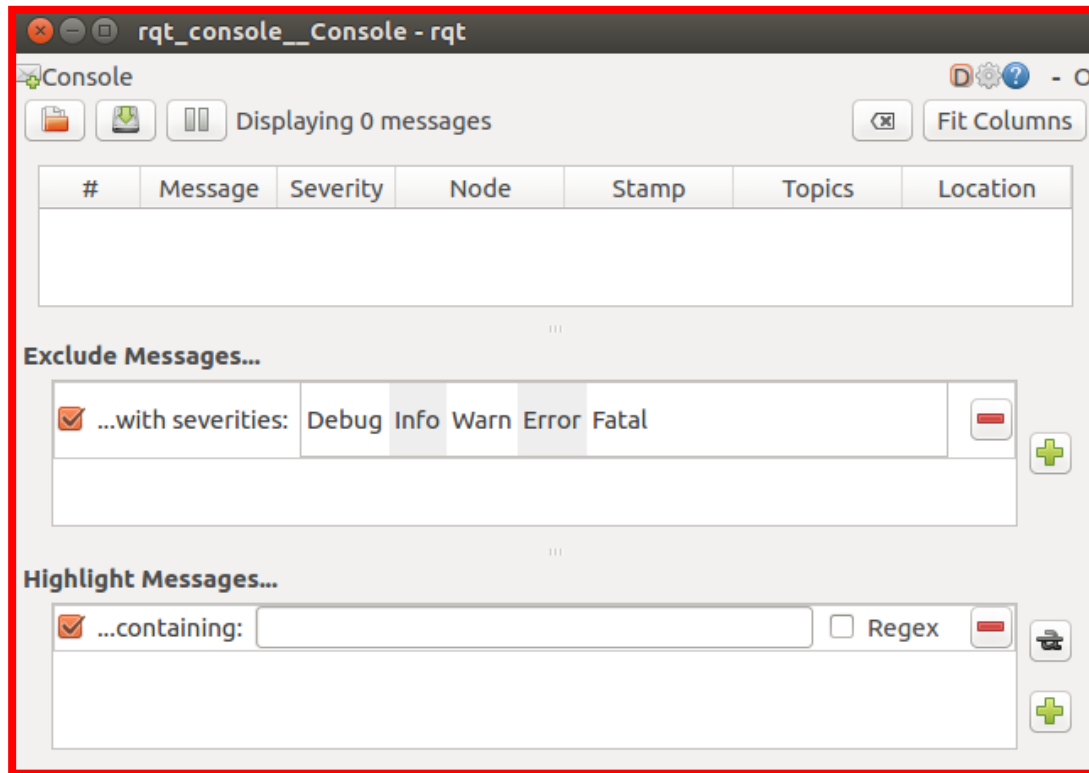


rqt\_console

的



rqt\_console



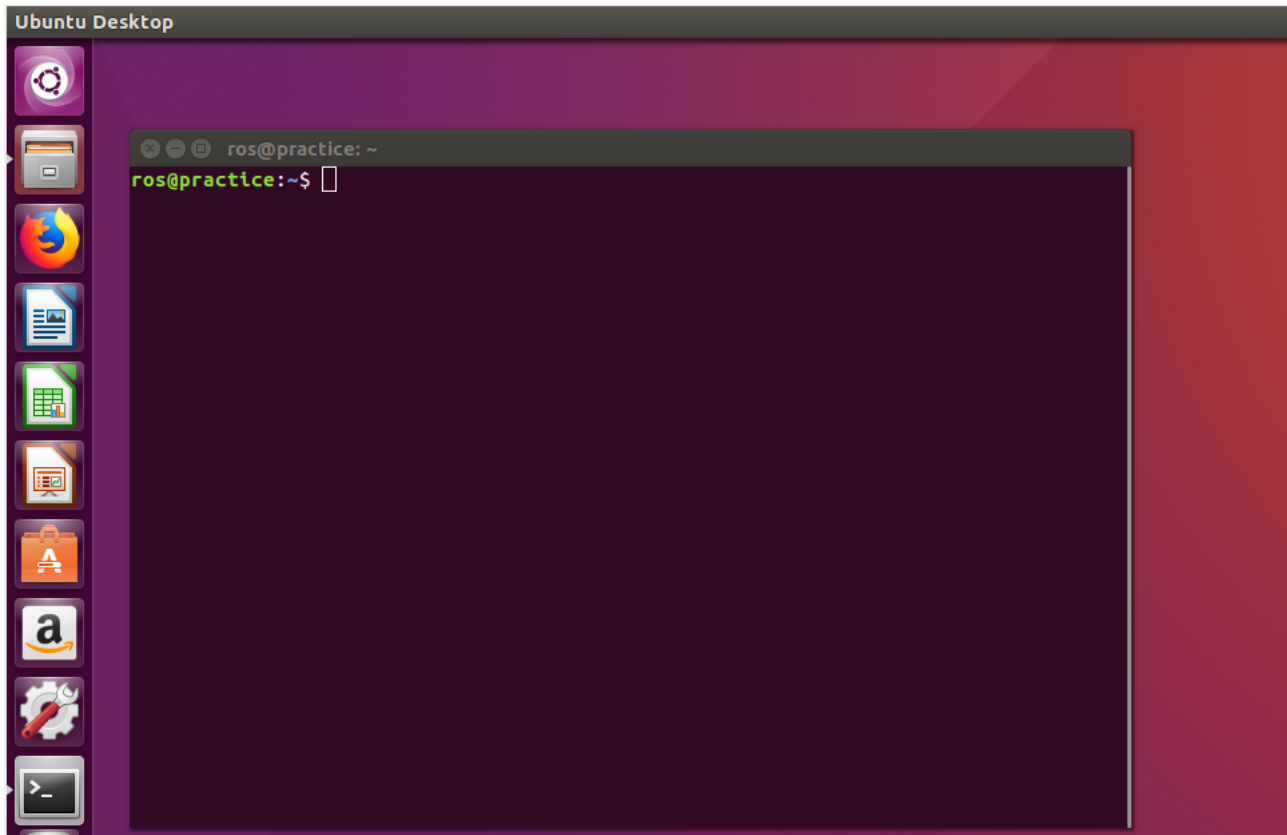
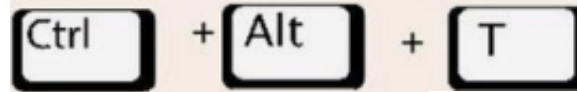
5-2  
rqt的console  
套件的API被  
開啟



## 操作 Turtlesim

## 查看ROS日誌紀錄

### 6.開啟新的 Terminal





## 操作 Turtlesim

## 查看ROS日誌紀錄

7.利用 rosrun 啟動位於



rqt\_logger\_level

的



rqt\_logger\_level

指令：`$ rosrun rqt_logger_level rqt_logger_level`

(藍色字體：package 的名稱、綠色字體：node 名稱)

利用



rqt\_logger\_level

調整 node 的 日誌(logging)訊息

```
ros@practice:~$ rosrun rqt_logger_level rqt_logger_level
```

7-1  
輸入指令



## 操作 Turtlesim

## 查看ROS日誌紀錄

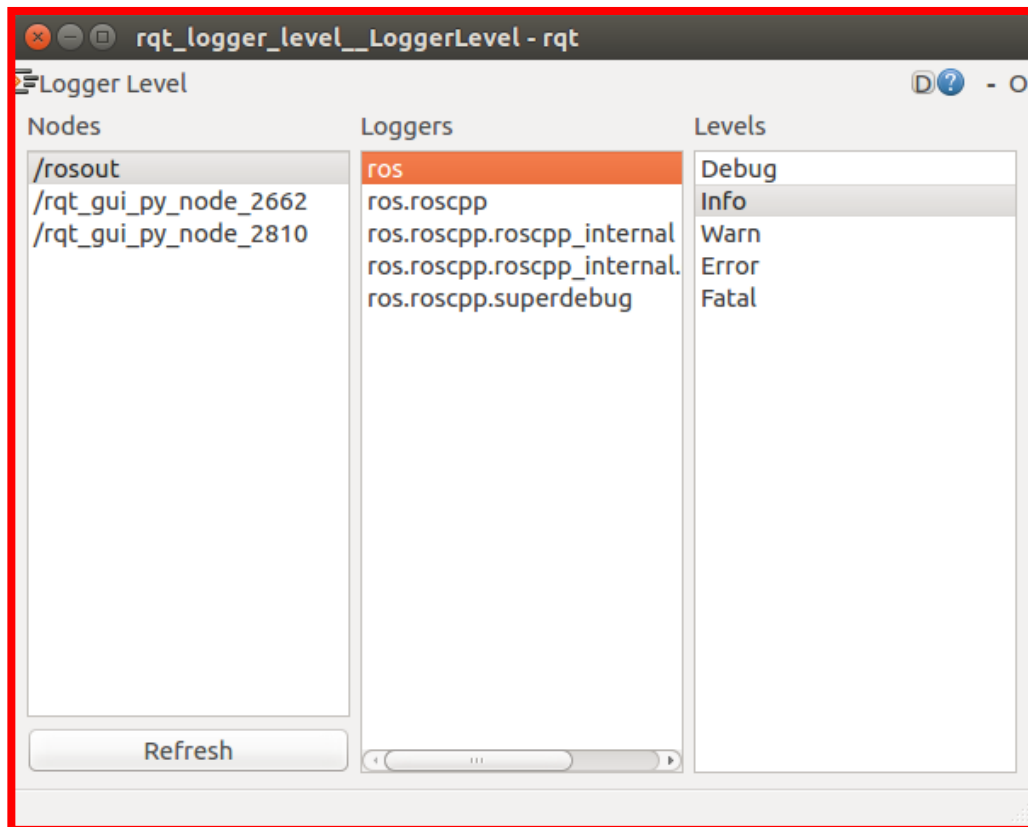
7.利用 rosrun 啟動位於



的



rqt\_logger\_level



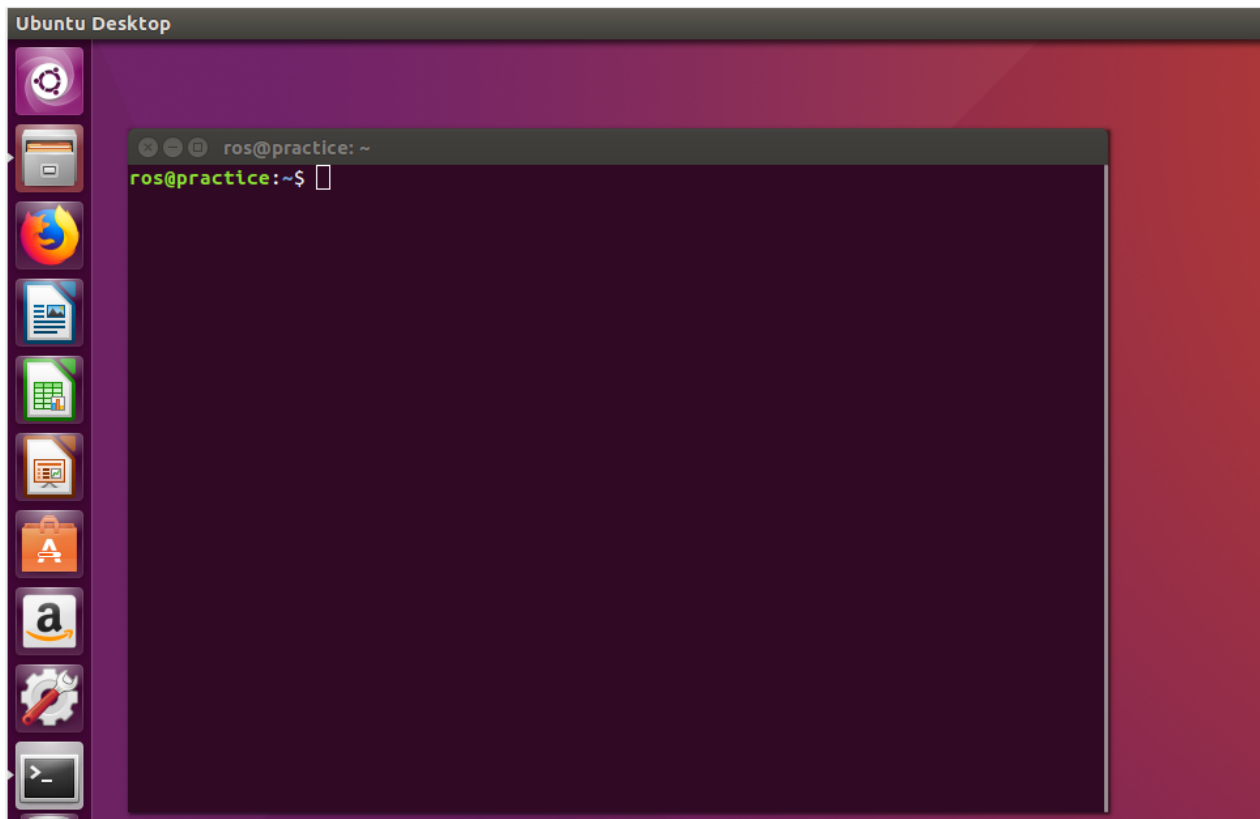
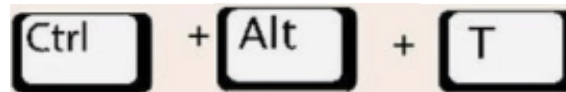
7-2  
rqt的  
logger\_level套  
件的API被開啟



## 操作 Turtlesim

## 查看ROS日誌紀錄

### 8.開啟新的 Terminal







## 操作 Turtlesim

## 查看ROS日誌紀錄

8.利用 rosrun 啟動位於



turtlesim

的



turtlesim\_node

指令：`$ rosrun turtlesim turtlesim_node`

藍色字體：package 的 名稱

綠色字體：node 名稱

```
ros@practice:~$ rosrun turtlesim turtlesim node  
[ INFO] [1520502221.562893958]: Starting turtlesim with node name /turtlesim  
[ INFO] [1520502221.575128309]: Spawning turtle [turtle1] at x=[5.544445], y=  
0.000000]
```

8-1

輸入指令



## 操作 Turtlesim

## 查看ROS日誌紀錄

## 8.利用 rosrun 啟動位於



turtlesim

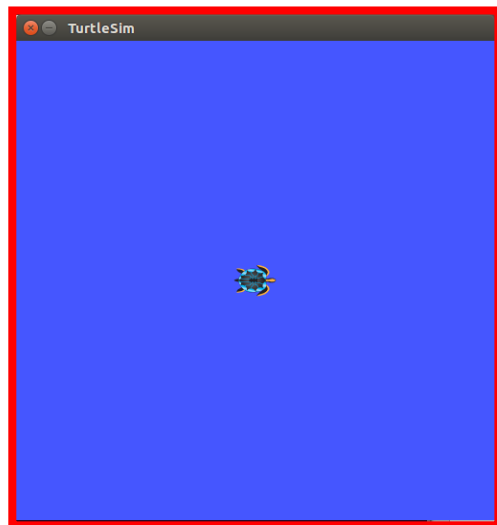
的



turtlesim\_node

```
ros@practice:~$ rosrun turtlesim turtlesim_node
```

```
[ INFO] [1520502221.562893958]: Starting turtlesim with node name /turtlesim  
[ INFO] [1520502221.575128309]: Spawning turtle [turtle1] at x=[5.544445], y=  
0.000000]
```



8-2



turtlesim\_node

啟動後資訊

8-3

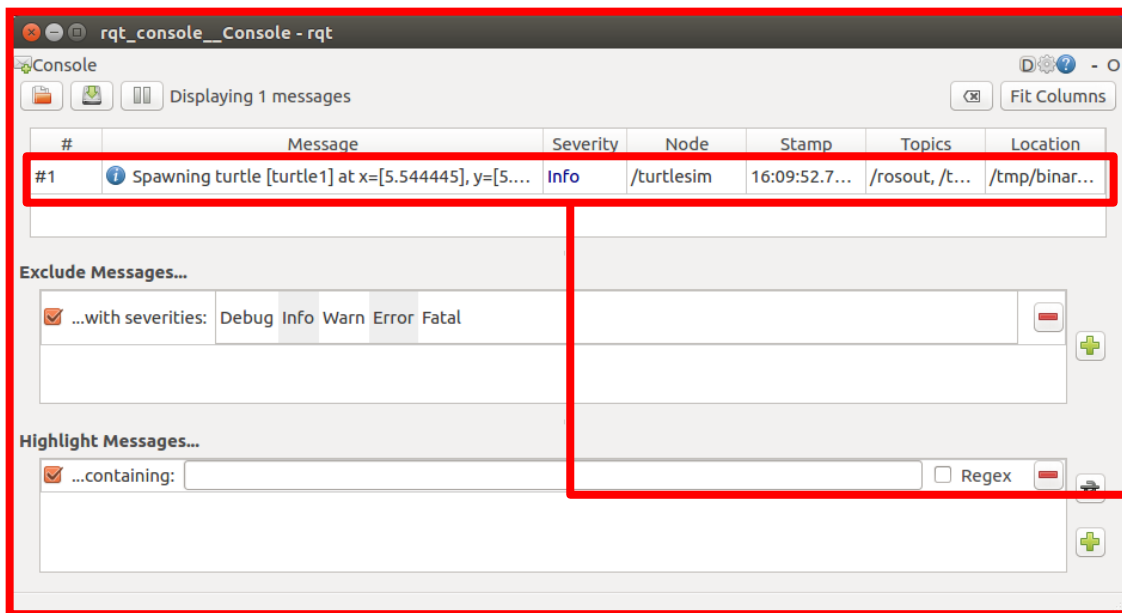
啟動後畫面  
(烏龜為隨機載入)



## 操作 Turtlesim

## 查看ROS日誌紀錄

## 9. 查看 ROS 的 node 日誌訊息



9-1  
移動到  
rqt\_console  
的視窗

9-2  
出現  
  
的logger 訊息



## 操作 Turtlesim

## 查看ROS日誌紀錄

## 10. 調整 Node 的 日誌等級(logger level)

level	logger
High	• Fatal → 可查看 Fatal、Error、Warn、Info、Debug 等級資料
	• Error → 可查看 Error、Warn、Info、Debug 等級資料
	• Warn → 可查看 Warn、Info、Debug 等級資料
	• Info → 可查看 Info、Debug 等級資料
Low	• Debug → 可查看 Debug 等級資料



## 操作 Turtlesim

## 查看ROS日誌紀錄

## 10. 調整 Node 的 日誌等級(logger level)

指令：

```
$ rostopic pub /turtle1/cmd_vel geometry_msgs/Twist -r 1 -- '{linear: {x: 2.0, y: 0.0, z: 0.0}, angular: {x: 0.0,y: 0.0,z: 0.0}}'
```

紅色字體：Topic 的名稱

紫色字體：Message 的名稱

綠色字體：Topic 發布的次數

淺澄字體：告知解析器之後的選項為參數

藍色字體：輸入的參數

```
ros@practice:~$ rostopic pub /turtle1/cmd_vel geometry_msgs/Twist -r 1 -- '{linear: {x: 2.0, y: 0.0, z: 0.0}, angular: {x: 0.0,y: 0.0,z: 0.0}}'
```

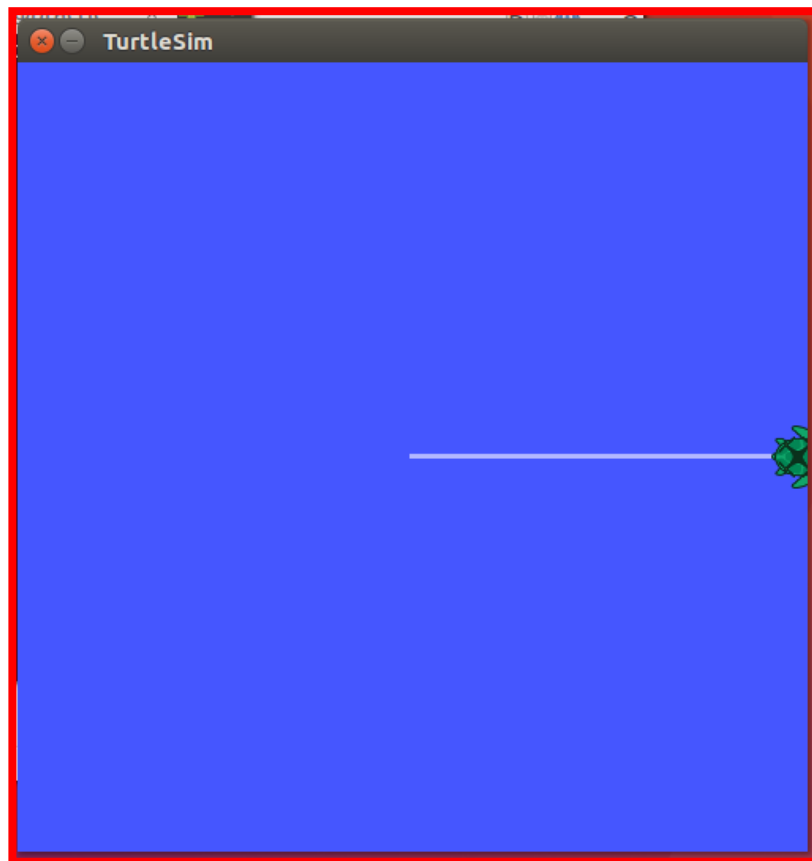
10-1  
輸入指令



## 操作 Turtlesim

## 查看ROS日誌紀錄

### 10. 調整 Node 的 日誌等級(logger level)



10-2

移到 Turtlesim視窗，  
發現烏龜一直撞牆

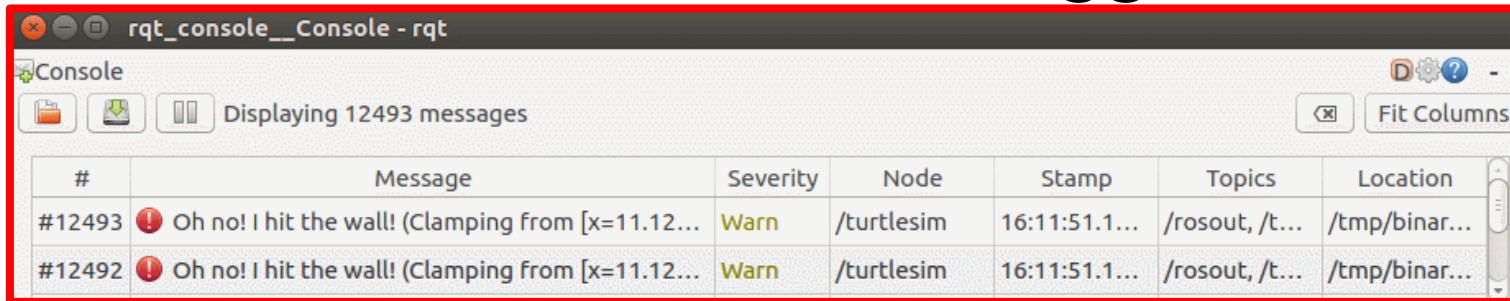




## 操作 Turtlesim

## 查看ROS日誌紀錄

## 10. 調整 Node 的 日誌等級(logger level)



rqt\_console\_\_Console - rqt

Console

Displaying 12493 messages

Fit Columns

#	Message	Severity	Node	Stamp	Topics	Location
#12493	Oh no! I hit the wall! (Clamping from [x=11.12...	Warn	/turtlesim	16:11:51.1...	/rosout, /t...	/tmp/binar...
#12492	Oh no! I hit the wall! (Clamping from [x=11.12...	Warn	/turtlesim	16:11:51.1...	/rosout, /t...	/tmp/binar...

```
99, y=5.544446])  
[ WARN ] [1521188143.507709514]: Oh no! I hit the wall! (Clamping from [x=11.1208  
99, y=5.544446])  
[ WARN ] [1521188143.523614866]: Oh no! I hit the wall! (Clamping from [x=11.1208  
99, y=5.544446])  
[ WARN ] [1521188143.538875985]: Oh no! I hit the wall! (Clamping from [x=11.1208  
99, y=5.544446])  
[ WARN ] [1521188143.554883086]: Oh no! I hit the wall! (Clamping from [x=11.1208  
99, y=5.544446])  
[ WARN ] [1521188143.571379245]: Oh no! I hit the wall! (Clamping from [x=11.1208  
99, y=5.544446])  
[ WARN ] [1521188143.586960134]: Oh no! I hit the wall! (Clamping from [x=11.1208  
99, y=5.544446])  
[ WARN ] [1521188143.603425664]: Oh no! I hit the wall! (Clamping from [x=11.1208  
99, y=5.544446])  
[ WARN ] [1521188143.619707666]: Oh no! I hit the wall! (Clamping from [x=11.1208  
99, y=5.544446])  
[ WARN ] [1521188143.635883524]: Oh no! I hit the wall! (Clamping from [x=11.1208  
99, y=5.544446])  
[ WARN ] [1521188143.6538875985]: Oh no! I hit the wall! (Clamping from [x=11.1208  
99, y=5.544446])  
[ WARN ] [1521188143.671881434]: Oh no! I hit the wall! (Clamping from [x=11.1208  
99, y=5.544446])
```

## 10-2

移到 rqt\_console 套件  
發現日誌訊息被更新

## 10-3

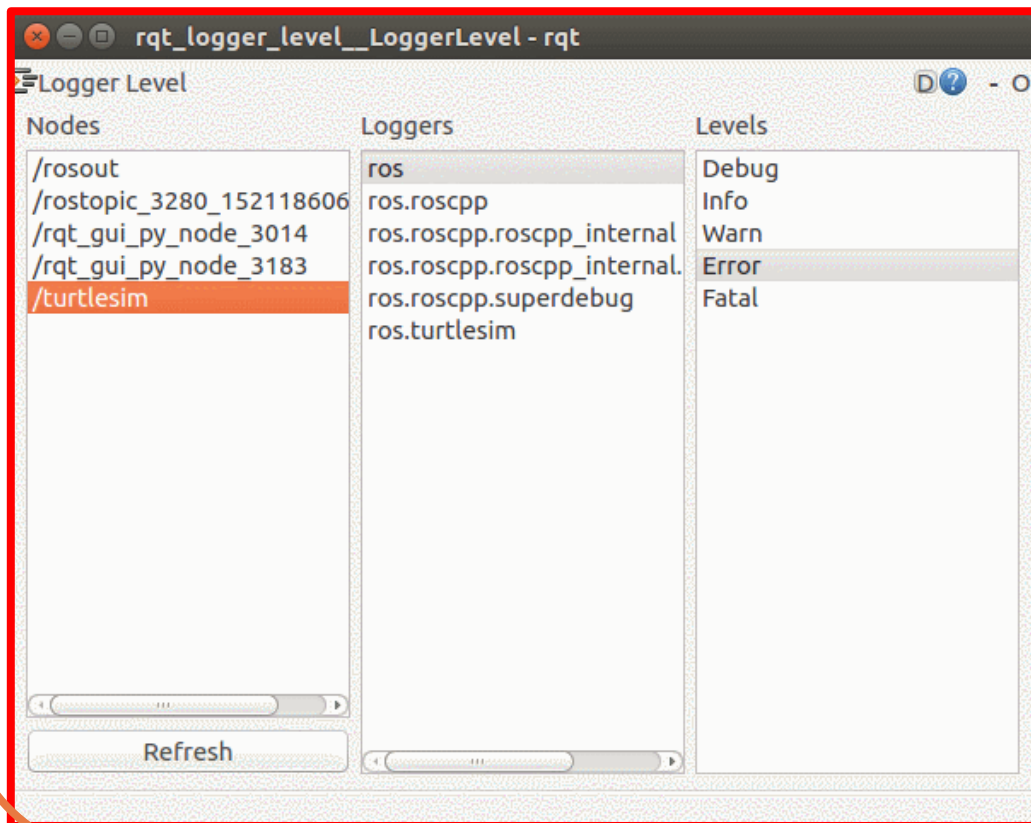
移到啟動 turtlesim\_node 的  
Terminal 發現  
訊息也被更新



## 操作 Turtlesim

## 查看ROS日誌紀錄

## 10. 調整 Node 的 日誌等級(logger level)



10-4

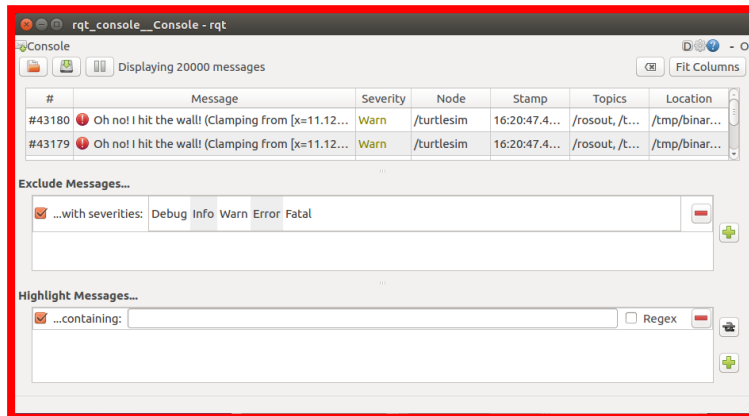
移動到 `logger_level`  
的 API 介面，並選擇  
Node → `/turtlesim`  
Loggers → `ros`  
Level → `Error`



## 操作 Turtlesim

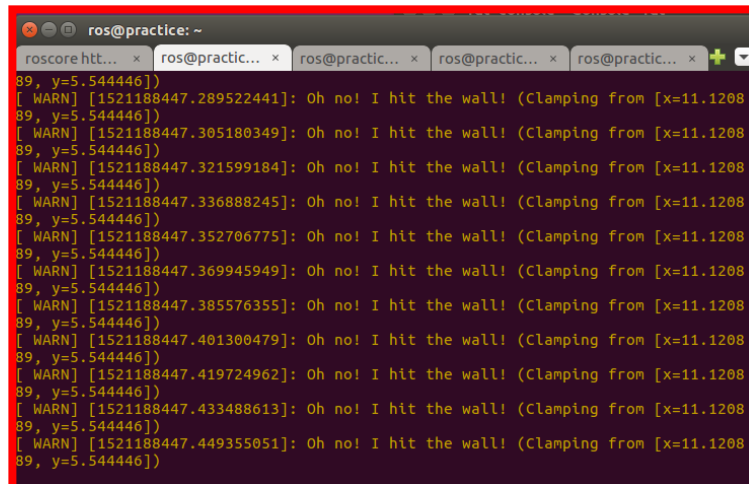
## 查看ROS日誌紀錄

## 10. 調整 Node 的 日誌等級(logger level)




10-5

移到 rqt\_console  
套件 API，發現日誌訊  
息不再被更新



10-3

移到啟動  的  
Terminal 發現  
日誌訊息不再被更新



## 操作 Turtlesim

## 執行多個節點

## 11. roslaunch 的使用方法

指令：

```
$ roslaunch package *.launch
```

藍色字體：package 的名稱

綠色字體：node 名稱



\*.launch

- 基於 XML 格式
- 描述如何啟動多個 Node



node1.py



node5.py



node6.py



node9.py

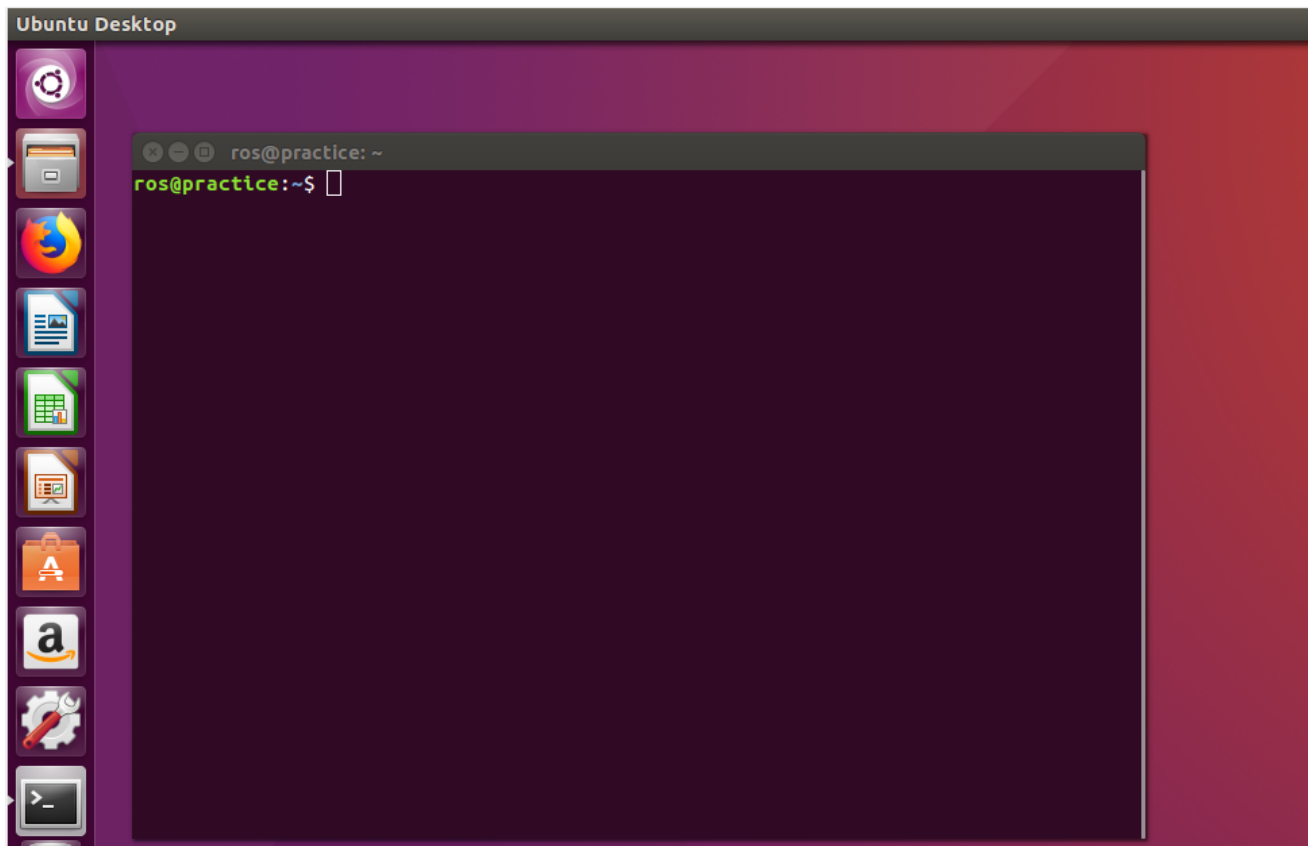
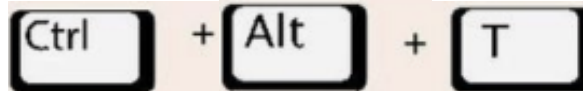




## 操作 Turtlesim

## 執行多個節點

### 12.開啟新的 Terminal





## 操作 Turtlesim

## 執行多個節點

13.在

Home

practice

catkin\_ws

src

first\_pkg



launch

指令：`$ cd ~/practice/catkin_ws/src/first_pkg`

將 Terminal 索引目錄至

Home

practice

catkin\_ws

src

first\_pkg

```
ros@practice:~$ cd ~/practice/catkin_ws/src/first_pkg/  
ros@practice:~/practice/catkin_ws/src/first_pkg$
```


13-1  
輸入指令13-2  
目前索引的目錄





## 操作 Turtlesim

## 執行多個節點

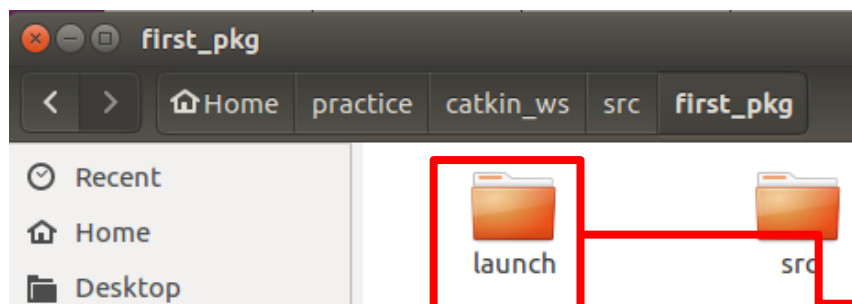
13.在  Home practice catkin\_ws src first\_pkg 建立

指令：\$ mkdir launch

建立



```
ros@practice:~/practice/catkin_ws/src/first_pkg$ mkdir launch
```

13-3  
輸入指令13-4  
資料夾建立完成



## 操作 Turtlesim

## 執行多個節點

14.在



建立



turtlemimic.launch

launch

Home practice catkin\_ws src first\_pkg launch

New Folder

New Document

Empty Document

Open in Terminal

Restore Missing Files...

Paste

Properties

14-1  
進入

14-2

在空白處按下滑鼠右  
鍵，並選擇  
「New Document」14-3  
選擇

「Empty Document」



## 操作 Turtlesim

## 執行多個節點

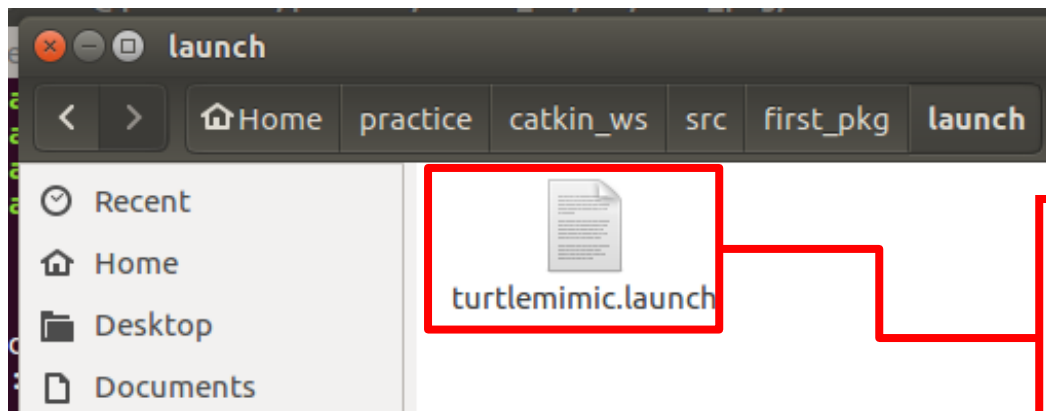
14.在 建立



launch



turtlemimic.launch



14-1

將檔案名稱命名為  
「turtlesim.launch」



## 操作 Turtlesim

## 執行多個節點

### 15.編輯



turtlemimic.launch

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

#### 15-1

將左方程式內容  
輸入至



turtlemimic.launch

並儲存檔案

程式來源 : <http://wiki.ros.org/ROS/Tutorials/UsingRqtconsoleRoslaunch>



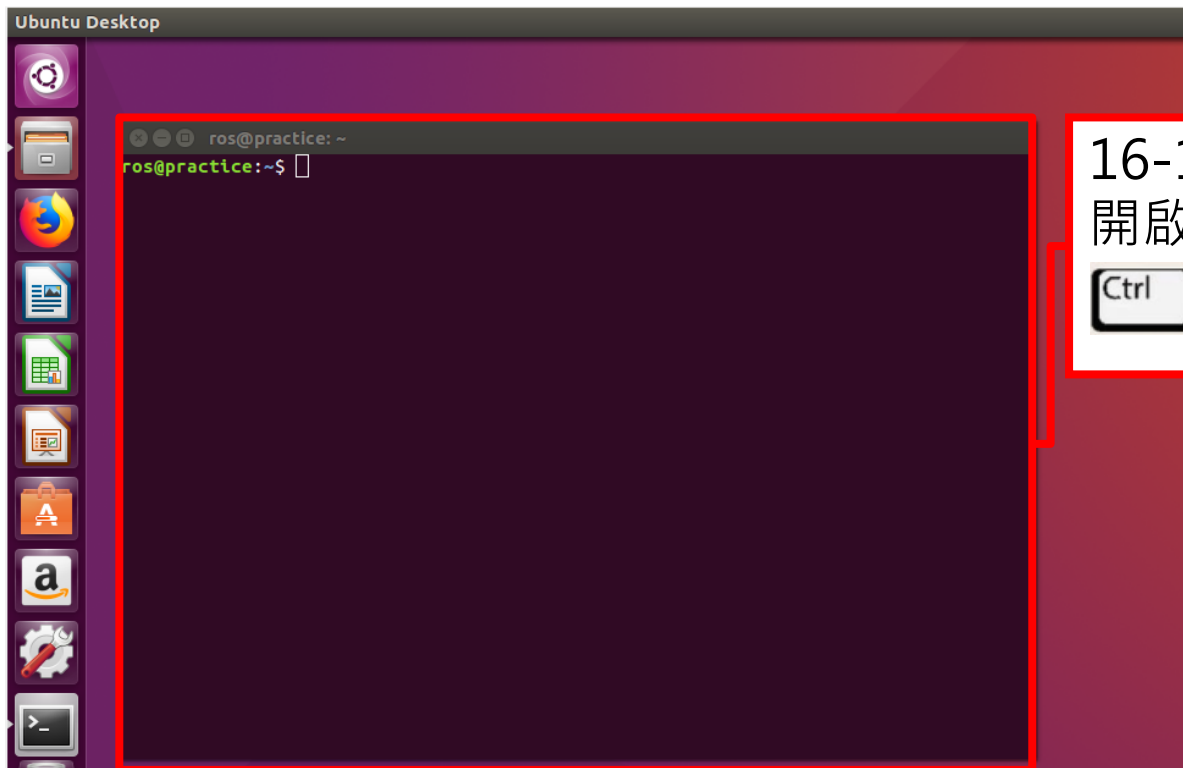
## 操作 Turtlesim

## 執行多個節點

### 16.啟動



turtle\_mimic.launch



16-1

開啟新的 Terminal

Ctrl + Alt + T



## 操作 Turtlesim

## 執行多個節點

### 16.啟動



turtlemimic.launch

路徑：

Home practice catkin\_ws src first\_pkg launch

指令：`$ roslaunch first_pkg turtlemimic.launch`

藍色字體：`package`的名稱

橙色字體：`launch` 檔案的名稱

```
ros@practice:~$ roslaunch first_pkg turtlemimic.launch
... logging to /home/ros/.ros/log/184a1640-2b1e-11e8-8e...
-practice-2835.log
Checking log directory for disk usage. This may take a while.
Press Ctrl-C to interrupt
Done checking log file disk usage. Usage is <1GB.
```

```
started roslaunch server http://practice:35729/
```

```
SUMMARY
=====
```

```
PARAMETERS
```

```
* /rostdistro: kinetic
* /rosversion: 1.12.12
```

#### 16-2

輸入指令

#### 16-3

若「Master Node」未啟動時，roslaunch指令會自動啟動「Master Node」





## 操作 Turtlesim

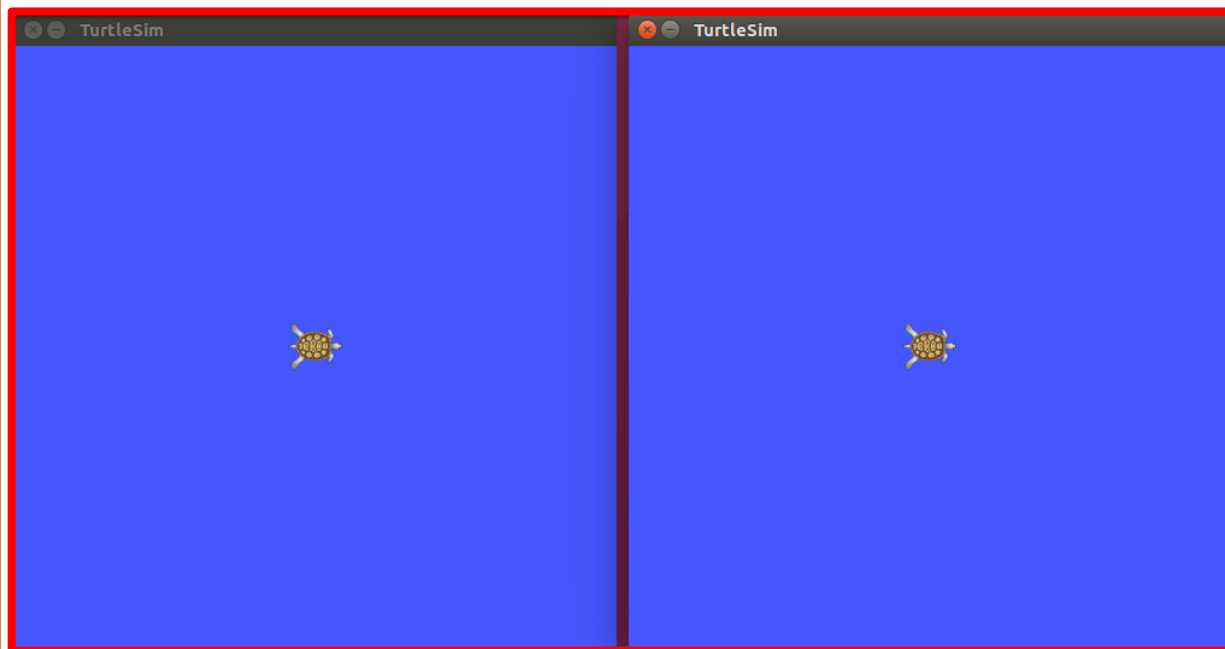
## 執行多個節點

16.啟動



turtlemimic.launch

路徑：

Home practice catkin\_ws src first\_pkg launch

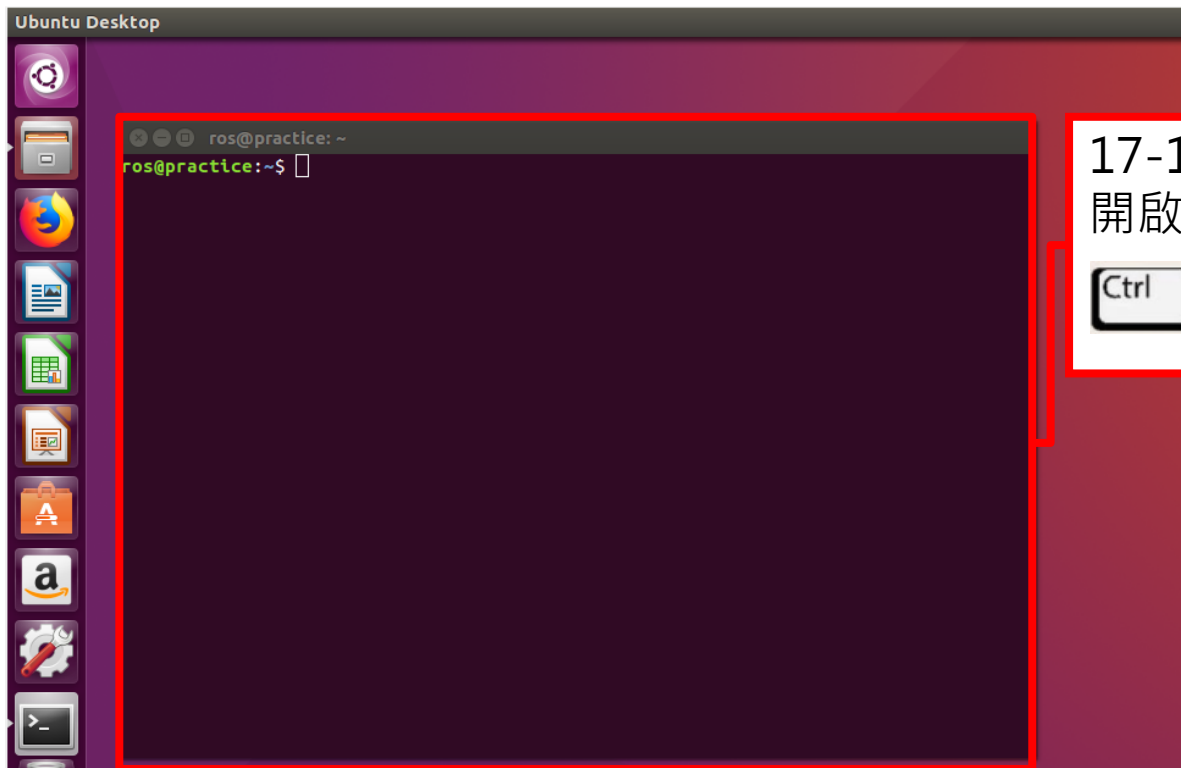
16-4  
發現  
ROS 啟動  
兩個 Node



## 操作 Turtlesim

## 執行多個節點

## 17. 使用 rostopic 發布 Topic



17-1

開啟新的 Terminal



\* 指令來源 : <http://wiki.ros.org/ROS/Tutorials/UsingRqtconsoleRoslaunch>



## 操作 Turtlesim

## 執行多個節點

## 17. 使用 rostopic 發布 Topic

指令：

```
$ rostopic pub /turtlesim1/turtle1/cmd_vel geometry_msgs/Twist -r 1 -- '[2.0, 0.0, 0.0]' '[0.0, 0.0, -1.8]'
```

綠色字體：Topic 發布的次數

紅色字體：Topic 的名稱

紫色字體：Message 的名稱

淺澄字體：告知解析器之後的選項為參數

藍色字體：輸入的參數

```
ros@practice:~$ rostopic pub /turtlesim1/turtle1/cmd_vel geometry_msgs/Twist  
-r 1 -- '[2.0, 0.0, 0.0]' '[0.0, 0.0, -1.8]'
```

17-2

輸入指令

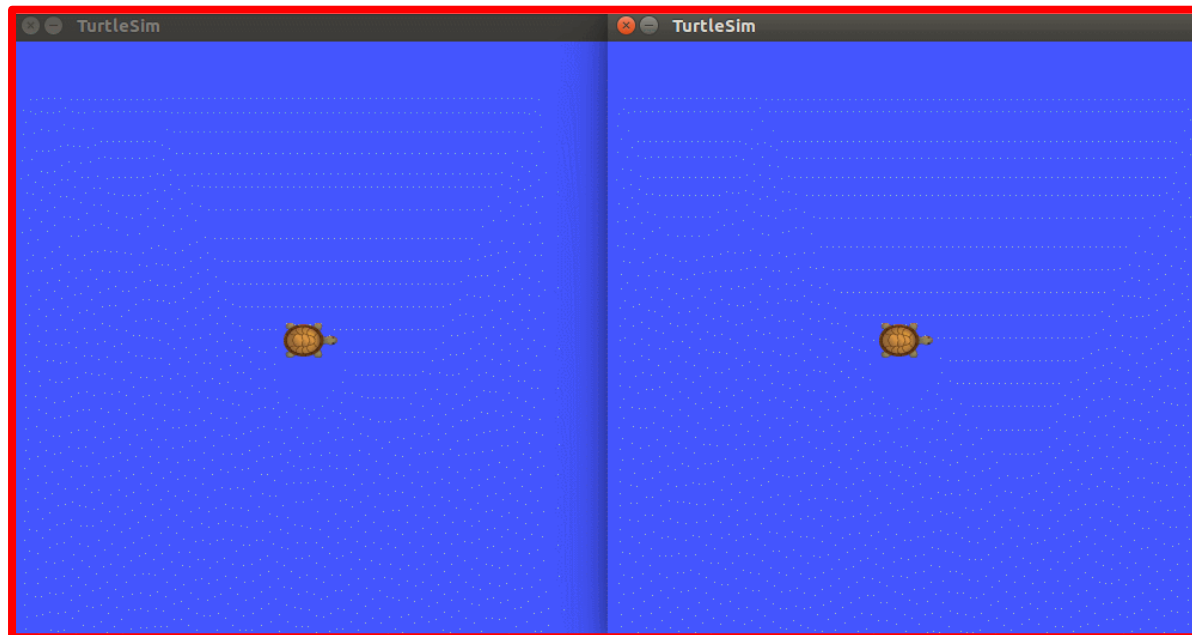
\* 指令來源：<http://wiki.ros.org/ROS/Tutorials/UsingRqtconsoleRoslaunch>



## 操作 Turtlesim

## 執行多個節點

## 17. 使用 rostopic 發布 Topic



17-3  
發現同一個  
Topic使兩個  
Node 移動

\* 指令來源 : <http://wiki.ros.org/ROS/Tutorials/UsingRqtconsoleRoslaunch>



# 操作 Turtlesim

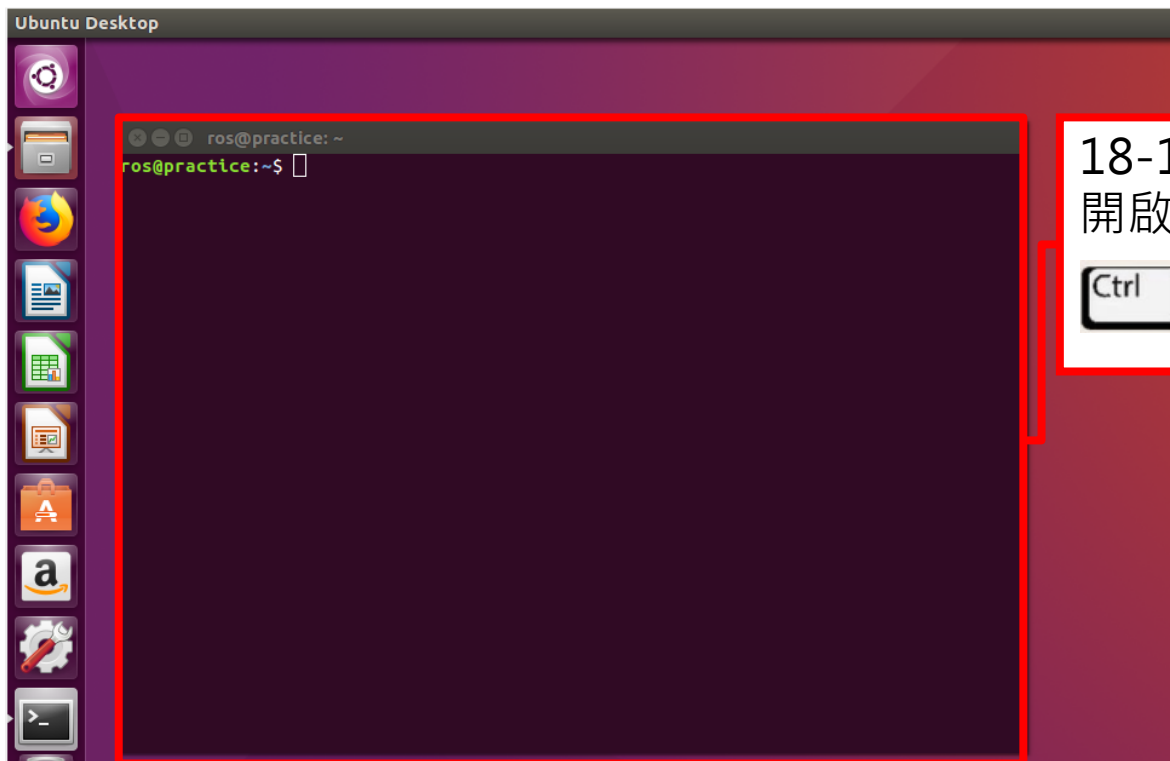
# 執行多個節點

## 18. 啟動



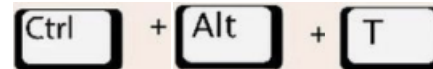
rqt\_graph

## 查看動態流程圖(dynamic graph)



18-1

開啟新的 Terminal



\* 指令來源 : <http://wiki.ros.org/ROS/Tutorials/UsingRqtconsoleRoslaunch>



## 操作 Turtlesim

## 執行多個節點

18. 啟動



查看動態流程圖(dynamic graph)

rqt\_graph

指令：`$ rosrun rqt_graph rqt_graph`

(藍色字體：package 的名稱、綠色字體：node 名稱)

利用



繪製ROS系統的動態流程圖(dynamic graph)

rqt\_graph

```
ros@practice:~$ rosrun rqt_graph rqt_graph
```

18-2

輸入指令





# 操作 Turtlesim

## 執行多個節點

### 18. 啟動



查看動態流程圖(dynamic graph)

rqt\_graph

開啟



查看程式內容對照動態流程圖

turtlemimic.launch

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



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



## 操作 Turtlesim

## 執行多個節點

18. 啟動



查看動態流程圖(dynamic graph)

rqt\_graph

開啟



查看程式內容對照動態流程圖

turtlemimic.launch



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