

# Bluetooth Tutorial

Larry Rudolph



Pervasive Computing MIT 6.883 SMA 5508 Spring 2006 Larry Rudolph

```
from bluetooth import *

target_name = "My Phone"
target_address = None

nearby_devices = discover_devices()

for address in nearby_devices:
    if target_name == Lookup_name( address ):
        target_address = address
        break

if target_address is not None:
    print "found target bluetooth device with address ",
    target_address
else:
    print "could not find target bluetooth device nearby"
```

# Server (rfcomm/L2CAP)

```
port = 1 #or 0x1001
```

```
server_sock=BluetoothSocket( RFCOMM) # or L2CAP  
server_sock.bind(("",port))  
server_sock.listen(1)
```

```
client_sock, client_info = server_sock.accept()  
print "Accepted connection from ", client_info
```

```
data = client_sock.recv(1024)  
print "received [%s]" % data
```

```
client_sock.close()  
server_sock.close()
```

# Service Discovery

```
port = get_available_port( RFCOMM )
```

```
server_sock=BluetoothSocket( RFCOMM )  
server_sock.bind(("",port))  
server_sock.listen(1)
```

```
advertise_service( server_sock, "Bluetooth Serial Port",  
                  service_classes = [ SERIAL_PORT_CLASS ],  
                  profiles = [ SERIAL_PORT_PROFILE ] )
```

```
client_sock, client_info = server_sock.accept()  
print "Accepted connection from ", client_info
```

```
data = client_sock.recv(1024)
```

```

import sys
from bluetooth import *

service_matches = find_service( name = "Bluetooth Serial
Port", uuid = SERIAL_PORT_CLASS )

if len(service_matches) == 0:
    print "couldn't find the service!": sys.exit(0)

first_match = service_matches[0]
port = first_match["port"]
name = first_match["name"]
host = first_match["host"]

print "connecting to ", host

sock=BluetoothSocket( RFCOMM )
sock.connect((host, port))
sock.send("hello!!")

```

## Dynamically allocate port

```

from bluetooth import *
socket = BluetoothSocket( RFCOMM )
while True:
    free_port = get_available_port( RFCOMM )
    try:
        socket.bind( ( "", free_port ) )
        break
    except BluetoothError:
        print "couldn't bind to ", free_port

# listen, accept, and the rest of the program...

```

```
from bluetooth import *
from select import *
```

## Asynchronous

```
class MyDiscoverer(DeviceDiscoverer):
    def pre_inquiry(self):
        self.done = False

    def device_discovered(self, address, device_class, name):
        print "%s - %s" % (address, name)

    def inquiry_complete(self):
        self.done = True

d = MyDiscoverer()
d.find_devices(lookup_names = True)

while True:
    can_read, can_write, has_exc = select( [d], [], [] )

    if d in can_read:
        d.process_event()

    if d.done: break
```

## If confused ...

- Can always go look at source ...
- on my linux machine,
  - /usr/lib/python2.3/site-packages/bluetooth.py
  - look at class DeviceDiscoverer for the skeleton code.