obfuscation nation

mobile application design

week 8: Obfuscators. HTTP POST. Bluetooth

michael sharon

Menu

- **Class outline**
- **Not the Phone of the Week**
- **Obfuscation in da Nation**
- **Tips’n’tricks: Graphics Edition**
- **Presentation: Jonathan + Fernando**
- **Uploading thangs with HTTP POST**
- **Bluetooth**

Class outline (old)

- Week 9: Playing (with) the future (pt.1) (November 2)
- Week 10: Playing (with) the future (pt.2) (November 9)
- Week 11: The hard cell: getting your stuff out there (November 16)
- Week 12: Final Project Workshop AKA Ze bug iz vere? (November 30)
- Week 13: Final Project Presentations (December 7)
- Week 14: Final Project Presentations (December 14)

Class outline (new)

- Week 9: Obfuscation, HTTP POST, Bluetooth (November 9)
- Week 10: Final Project Workshop AKA Open Questions / Issues (November 10)
- Week 11: Introduction to Python S60 (November 16)
- Week 12: WAP + XHTML MP (November 30)
- Week 13: Final Project Presentations (December 7)
- Week 14: Final Project Presentations (December 14)

Not the Phone of the Week AKA http://vertu.com

Obfuscation in da Nation

Obfuscation

- AKA bytecode obfuscator
- What is it?
  - Software build tool
- What does it do?
  - Reduces the size of your class files
  - Makes it difficult to decompile your code
  - Removes unused/unnecessary methods

Shrink, optimize, obfuscate

- Detect and remove unused classes, fields, methods, and attributes.
- Optimize bytecode and remove unused instructions.
- Rename the remaining classes, fields, and methods using short meaningless names.
- Resulting jars are smaller and harder to reverse-engineer.

*(thanks Proguard docs!)

Smaller JAR files

= reduced storage requirements
= faster downloading of your application
= faster loading
& smaller memory footprints

3 Obfuscators

- **Proguard**
  - http://proguard.sourceforge.net/
- **Retroguard**
  - http://www.retrologic.com/
- **Jode**
  - http://jode.sourceforge.net/

Tips'n'tricks: Graphics edition

Images/Icons

- PNG
  - Lossless competitor to GIF
  - Supports >256 colours and alpha
  - Only use indexed images! 8-bit or 4-bit
- Some support for transparency

Resolutions

- Nokia Series 60 3rd Edition = 352x216 - definitely not traditional
- Dual portrait + landscape modes

Resolutions

- 176w x variable height (mostly 208px) = current standard
  - Portrait orientation
  - Nokia Series 60 2nd Edition, Motorola RAZR, ROKR, PEBL, iDEN, Sony Ericsson
- 128w x variable height
  - Old standard
- 96x - even older, painful to resize artwork

Tools

- Photoshop / Gimp / Pixel
- Debabelizer
- PNGCrush, PNGOut, TweakPNG
- Icon Studio

Presentation: Jonathan + Fernando

HTTP Post

POST Process

1. Get yer **HttpConnection**
2. Add header fields - setRequestMethod()
3. Open the output stream (**openOutputStream()** - sends headers
4. Format + send parameters + request body
5. Read the response

Bluetooth

Bluetooth - JSR 82

- Java API for Bluetooth Wireless Technology
- Consists of two separate APIs

1. Java API for Bluetooth
    - Communications via BT radio
2. Java API for OBEX
    - The OBEX protocol

- **Optional API**

What is Bluetooth?

- Short range wireless networking
- Operates in the **2.4 Ghz ISM band**
- Named after a swedish King - Harald Blatand
- Cable replacement / short range serial
- Personal Area Network (PAN)

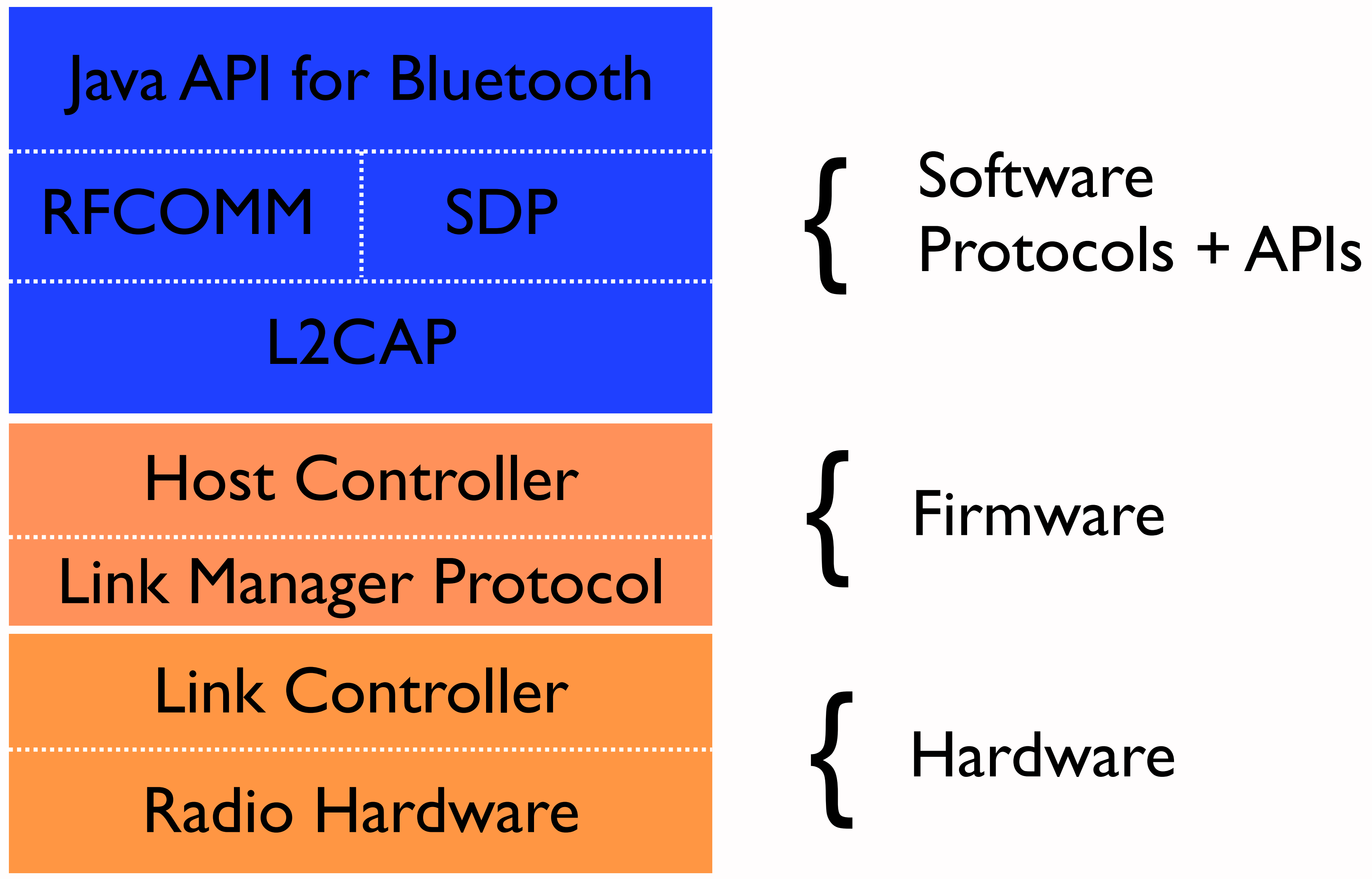
Bluetooth Info

- Bluetooth 1.0 + 1.1 = 1Mbit/s - only 720kb for user applications
- Bluetooth 2.0 supports EDR (Enhanced Data Rate) allows for up to 3Mb of raw data (2.1 Mb for your apps)
- Much lower in practice ~ 100kb/s

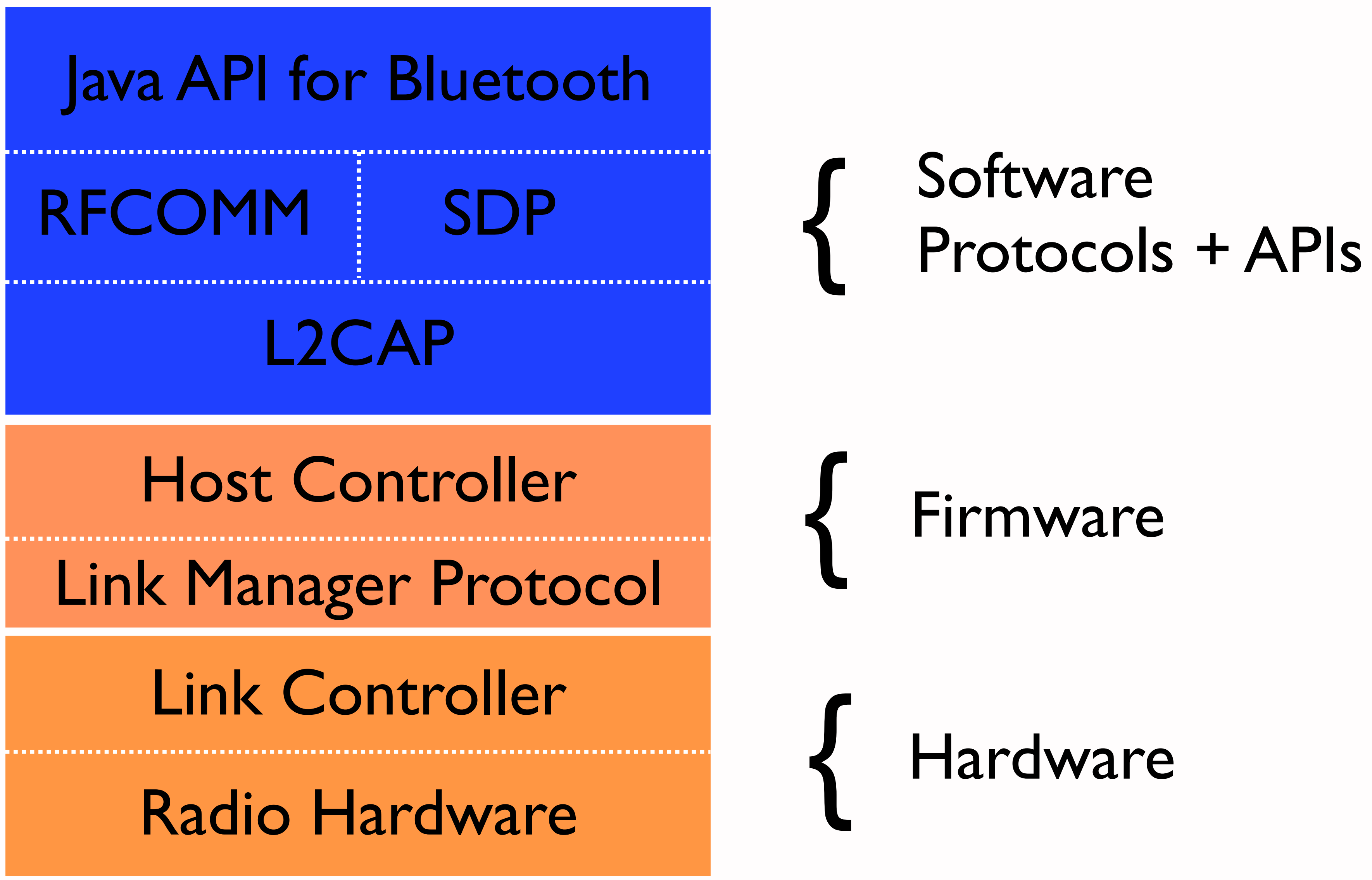
Useful links

- Official Bluetooth Site (lots of technical info)
  - http://bluetooth.com
- Java Bluetooth
  - http://javabluetooth.com
- JSR-82 Specification
  - http://jcp.org/en/jsr/detail?id=82

Bluetooth Implementation



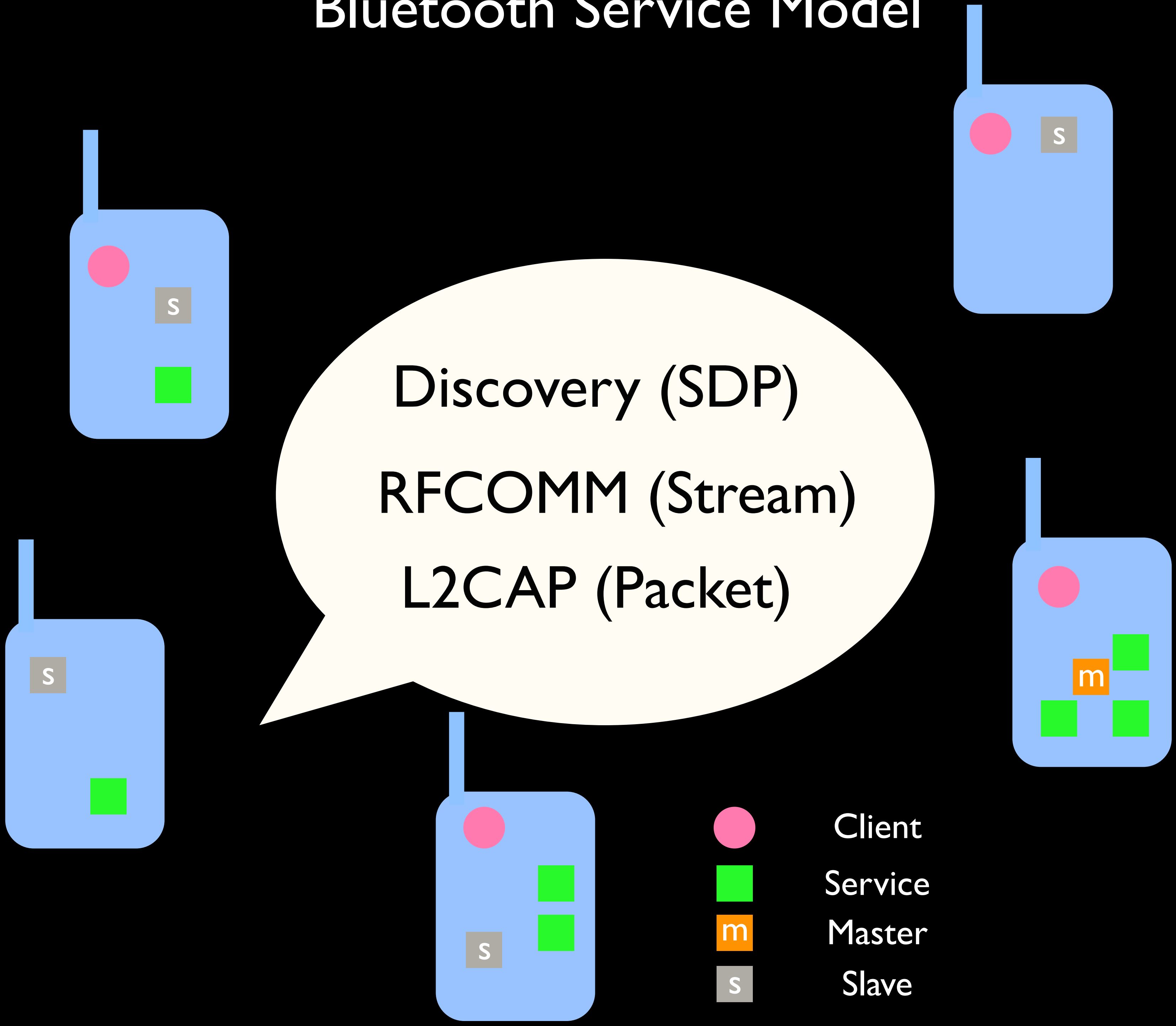
Bluetooth Implementation



Bluetooth Interactions

1. Discovery
2. Client activities / services
3. Server activities
4. Peer activities

Bluetooth Service Model



●	Client
🟩	Service
m	Master
s	Slave

Your BT Stack

- Local BT Stack accessed with LocalDevice
  - LocalDevice myDevice = LocalDevice.getLocalDevice()
- DiscoveryAgent.getFriendlyName()
- Methods
  - int getDiscoverable()
  - boolean setDiscoverable(int mode)
  - static String getProperty(String property)

Properties

- bluetooth.api.version
- bluetooth.master.switch
- bluetooth.sd.attr.retrievable.max
- bluetooth.connected.devices.max
- bluetooth.l2cap.receiveMTU.max
- bluetooth.sd.trans.max

Steps to connect

1. Access Local Device
2. Discover Devices (perform inquiry)
3. Discover Services (identified by UUID)
4. Connect with specific device / service