Hardware
- Main features
- Core platforms examples

Software
- Software architecture
- Operating systems
- Application development platforms

Applications
- Main applications

MediaArt
- Experiments using mobile devices
Hardware > Main features

- Flip upper & Flip lower
- LCD module & Sub board
- Main upper case
- Main board
Hardware > Main features > Flip upper & Flip lower

1. Flip deco
2. Sub window
3. Camera window
4. Reflection mirror
5. EMI paint
Hardware > Main features > LCD module & Sub board

1. Speaker (Speaker + Receiver)
2. Static electricity protection tape
3. Sub LCD
4. FPCB (Flexible PCB)
   - Connect LCD to Main board
5. Camera
   - CCD or CMOS

1. ASIC
   - Image capture and decoding etc.
   - Camera related process
2. FPCB
3. DSP
   - Compress image and send to LCD
Hardware > Main features > Main cases

Case

Motor (Vivrator)

* Main Upper

* Main lower

Metal dom (sense the press)

5 way navigation

0~9, *, # keypad
1. Ear mic jack
2. RF jack
   - RF verification jack after development
3. Antenna contact
4. Duplexer
   - Sort TX/RX from antenna
5. Power amp
   - Amplify signal
6. IF SAW filter
   - Filter noise and sent signal to IFR
7. RFT
   - Transmitter
8. IFR
   - Receiver
9. Sound chip
   - Yamaha is the most popular sound chip
10. MSM chip (Qualcomm)*
    - Main processor
11. Memory
    - ROM + RAM (recent: more than 200MB)
12. Mic
13. I/O Connector
    - Data communication
14. Switch
    - Side key connector
Example (Qualcomm MCU)

Radio Frequency module

http://www.cdmatech.com/products/msm7600_chipset_solution.jsp
Example (TI MCU)

Radio Frequency module

GSM only

Hardware > Core platforms > Example 2

http://focus.ti.com/general/docs/wtbu/wtbuportal.tsp?templateId=6123&contentId=4646
Software > Software architecture

Interactive environment | Mobile development

- Frameworks
  - UI / multimedia / fonts
- OS kernel & core APIs
- Chipset hardware platform
- Device drivers
- Peripheral devices

App. Frameworks
- Java
- Flash
- Browser
- Telephony
- File manager
- Settings
- PIMs
- Messaging
- Push Email
- Text editor
- Media Suite
- ...

Core Applications
- ...

Other Applications
- ...

Other Applications
- ...

App. Frameworks
Operating System

- More various than PC’s OS

- Symbian / Palm OS / Windows Mobile / OS X

- Closed and proprietary OS environment: very limited custom software development

- Runtime environments provided: JVM (Java Virtual Machine), SWF (ShockWave Flash), etc
1. **symbian**

- [http://www.symbian.com](http://www.symbian.com)

- Owned by Nokia etc.

- Offers APIs, user interface frameworks and reference implementations of common tools

- Default UI: **Nokia Series 60, Series 80, UIQ(User Interface Quartz)**

- Mainly C++ preference, but can also be programmed in OPL, Python, Visual Basic, Simkin, and Perl - together with the Java ME and PersonalJava flavours of Java.

- Microkernel architecture
  > contains scheduler, memory management, Bluetooth, IrDA, USB and UI APIs.
  > not contains networking, filesystem support (should be provided by user side servers)

- Flexibility but much vendor side integration works

- **Symbian phones**: Nokia, Foma series (Japan), Soni ericsson, Samsung, etc.
2. Palm OS

- http://www.access-company.com

- Originally released in 1996

- Mainly for PDAs

- Combined with a suite of basic applications including an address book, clock, note pad, sync, memo viewer and security software

- Primarily coded in C/C++

- Provides development tools: CASL (Compact Application Solution Language), AppForge Crossfire (C# based), Handheld Basic, HB++ (Visual basic based)

- Java runtime environment, Plua (a version of Lua) are available
3. **Windows Mobile**

- http://www.microsoft.com/windowsmobile/default.mspx

- PocketPC 2002 (powered by Windows CE 3.0)

- Windows Mobile 2003 (the 3rd version of PocketPC)

- Based on the Microsoft Win32 API

- Similar to desktop versions of Windows: Start button, Excel, PowerPoint, Outlook, Windows Media Player and etc
4. Linux

- [http://www.linuxdevices.com/](http://www.linuxdevices.com/)
- Open source
- Small footprint (around 2MB for a minimal installation)
- Motorola, Panasonic, NEC, Samsung, Telepong, Wildseed etc. adopted linux
5. OS X

All the power and sophistication of the world’s most advanced operating system — OS X — is now available on a small, handheld device that gives you access to true desktop-class applications and software, including rich HTML email, full-featured web browsing, and applications such as widgets, Safari, calendar, text messaging, Notes, and Address Book. iPhone is fully multi-tasking, so you can read a web page while downloading your email in the background. This software completely redefines what you can do with a mobile phone. – from www.apple.com
## Overview

<table>
<thead>
<tr>
<th>Platform</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Symbian</strong></td>
<td>Strongly supported by Nokia with waning support from other device makers. Currently large device deployments in Europe, with little penetration in the US market.</td>
</tr>
<tr>
<td><strong>Java ME</strong></td>
<td>Ideal for an all-around solution, if the Java ME platform provides the needed functionality.</td>
</tr>
<tr>
<td><strong>Python</strong></td>
<td>Ideal for initial prototyping and concept testing when functionality falls outside Java ME.</td>
</tr>
<tr>
<td><strong>Flash Lite</strong></td>
<td>Ideal for Graphics-heavy options with a market that can support the Flash Lite player.</td>
</tr>
<tr>
<td><strong>.NET Compact Framework</strong></td>
<td>Ideal for deployment on homogeneous Pocket-PC devices.</td>
</tr>
<tr>
<td><strong>Microbrowser Based</strong></td>
<td>Ideal for lightweight functionality, a web-interface for an existing application with no latency concerns, or a widely varying platform base</td>
</tr>
<tr>
<td><strong>BREW</strong></td>
<td>Ideal for deploying applications for deployment on CDMA-based networks with a deployed Brew Content Platform especially if OTA app deployment is desired.</td>
</tr>
<tr>
<td><strong>Pocket PC &amp; Windows Mobile</strong></td>
<td>Ideal for enterprise applications with an existing PC infrastructure and options for significant development investment.</td>
</tr>
<tr>
<td><strong>Palm OS</strong></td>
<td>Significant player with strong enterprise following in the important US market. PalmOS makes up the lions share of revenues for most consumer focused developers.</td>
</tr>
</tbody>
</table>
## Comparison

<table>
<thead>
<tr>
<th>Foundation</th>
<th>Learning Curve</th>
<th>Debuggers</th>
<th>Emulator available</th>
<th>IDE</th>
<th>Cross-Platform Deployment</th>
</tr>
</thead>
<tbody>
<tr>
<td>Symbian</td>
<td>C++</td>
<td>Good</td>
<td>Free Emulator</td>
<td>Many choices</td>
<td>Compile per target</td>
</tr>
<tr>
<td>Java ME</td>
<td>Java</td>
<td>Excellent</td>
<td>Free Emulator, Sun Java Wireless Toolkit, mpowerplayer</td>
<td>Eclipse, NetBeans Mobility Pack, Processing</td>
<td>Average*6)</td>
</tr>
<tr>
<td>Python</td>
<td>Python</td>
<td>Average</td>
<td>Add-on to Nokia Emulator</td>
<td>Several, plugins for Eclipse</td>
<td>Interpreted language only on Nokia Series60</td>
</tr>
<tr>
<td>Flash Lite</td>
<td>ActionScript</td>
<td>Average</td>
<td>Good</td>
<td>Macromedia Flash MX2004/8, Eclipse</td>
<td>Excellent *7)</td>
</tr>
<tr>
<td>.NET Compact Framework</td>
<td>C#, VB.NET</td>
<td>Average</td>
<td>Excellent</td>
<td>Visual Studio 2005, 2003</td>
<td>Windows Mobile</td>
</tr>
<tr>
<td>Microbrowser</td>
<td>XHTML (WAP 2.0)</td>
<td>Varies by Server-side scripting language</td>
<td>Good</td>
<td>Many</td>
<td>Many</td>
</tr>
<tr>
<td>BREW</td>
<td>C</td>
<td>Difficult *2)</td>
<td>No Debugger *4)</td>
<td>No Emulator</td>
<td>Visual Studio 6.0, Visual Studio 2003 .net</td>
</tr>
<tr>
<td>Pocket PC</td>
<td>C, C++</td>
<td>Average</td>
<td>Excellent</td>
<td>Visual Studio 2005</td>
<td>Windows Mobile</td>
</tr>
<tr>
<td>Palm OS</td>
<td>C, C++</td>
<td>Excellent</td>
<td>Average</td>
<td>Free Emulator</td>
<td>Eclipse, CodeWarrior</td>
</tr>
</tbody>
</table>

*1) Unusual C++ APIs, poor debugger support, and Symbian 9 breaks binary compatibility  
*2) But easier, and less featureful, than Symbian  
*3) Excellent for Win32 developers  
*4) Can use Visual Studio to debug the x86 testing code  
*5) Has a simulator for the x86 testing code  
*6) Many VM implementations have device specific bugs necessitating separate builds  
*7) Bundled – Top 5 mobile manufacturers, limited handset model support as of 3/06, best web compatibility  

Sun, Java wireless toolkit

JavaME developer's home: http://java.sun.com/javame/index.jsp
Mobile processing

- JavaME based development environment
- APIs for graphics, connectivity
- JavaME emulator
- Familiar to processing users

Mobile processing: [http://mobile.processing.org](http://mobile.processing.org)
Development environment

Mobile processing: [http://mobile.processing.org](http://mobile.processing.org)
Cellphone oriented libraries

Phone
This class provides access to the phone-specific features of the Phone library.

Phone
vibrate()
flash()
call()
lunch()
fullscreen()
noFullscreen()

Messenger
This class provides the primary interface for sending and receiving messages.

Messenger
send()
EVENT MSG RECEIVED

Message
This class represents messages that have been received.

Message
readBytes()
readString()
Cellphone oriented libraries

Bluetooth
This class provides the primary interface for discovering and establishing a Bluetooth network connection.

Bluetooth
cancel()
discover()
find()
start()
stop()

Device
Objects of this class represent nearby devices discovered on the Bluetooth network.

Device
name
address
cancel()
discover()

Service
Objects of this class represent software running on devices that can be connected to via the Bluetooth network.

Service
name
description
provider
device
connect()

Client
Client objects are used to communicate with other devices and services.

Client
read()
readBoolean()
readBytes()
readChar()
readInt()
readUTF()
skipBytes()
stop()
write()
writeBoolean()
writeChar()
writeInt()
writeUTF()
Cellphone oriented libraries

**MAudio3D**
Library for 3D audio.

**M3D**
Library for 3D graphics.

**MBluetooth**
Library for Bluetooth communication.

**MClientServer**
Library for creating client/server connections. Used with MBluetooth for creating discoverable services.

**MLocation**
Library for location positioning.

**MMessaging**
Library for sending and receiving text messages.

**MNRCode**
Library for decoding QR Code 2D barcodes.

**MPush**
Library for supporting push registry activation of applications (timer or connection activated).

**MSound**
Library for sound playback and recording.

**MVideo**
Library for video playback.

**MWebServices**
Library for consuming web services.
Macromedia FlashLite

- Lightweight version of Adobe Flash Player optimized for mobile phones and other devices

- Powerful graphic expression

- Support the W3C Standard SVG Tiny

- Presets for specific models of Nokia, DoCoMo etc. on Flash development

- Flash: http://adobe.com
- Flash Mobile and Device Seminar: https://admin.adobe.acrobat.com/_a227210/p75414944/?trackingid=CXND
Development environment

- Flash: [http://adobe.com](http://adobe.com)
- Flash Mobile and Device Seminar: [https://admin.adobe.acrobat.com/_a227210/p75414944/?trackingid=CXND](https://admin.adobe.acrobat.com/_a227210/p75414944/?trackingid=CXND)
Cellphone oriented libraries

- Flash Lite Specific Language Elements
  - Capabilities
    - _capCompoundSound
    - _capEmail
    - _capLoadData
    - _capMFI
    - _capMIDI
    - _capMMS
    - _capMP3
    - _capSMAF
    - _capSMS
    - _capStreamSound
    - _cap4WayKeyAS
    - $version
  - Iscommand()
    - Launch
  - Iscommand2()
    - Escape
    - FullScreen
- GetBatteryLevel
  - GetDeviceID
  - GetFreePlayerMemory
  - GetLanguage
  - GetLocaleLongDate
  - GetLocaleShortDate
  - GetLocaleTime
  - GetMaxBatteryLevel
  - GetMaxSignalLevel
  - GetMaxVolumeLevel
  - GetNetworkConnectStatus
  - GetNetworkName
  - GetNetworkRequestStatus
  - GetNetworkStatus
  - GetPlatform
  - GetPowerSource
  - GetSignalLevel
  - GetTimeHours
  - GetTimeMinutes
  - GetTimeSeconds
  - GetTimeZoneOffset
  - GetTotalPlayerMemory
  - GetVolumeLevel
  - Quit
  - ResetSoftKeys
  - SetInputTextType
  - SetQuality
  - SetSoftKeys
  - StartVibrate
  - StopVibrate
  - Unescape
Overall

- Telephony
  Voice call / Video call / Conference call..

- Messaging
  Voice message / Text message / Multimedia message / Email / PTT..

- Media Suite
  Camera / Video / media player / MIDI / MP3 player..

- PIM
  Organizer / Contacts / Alarm / Note..

- Browser
  WAP / Web browser

- Connectivity
  IrDA / Bluetooth / GPS
Telephony

Diagram showing the components of a mobile telephony system, including SIM (Subscriber Identity Module), ME (Mobile Equipment), BTS (Base Transceiver Station), BSC (Base Station Controller), HLR (Home Location Register), VLR (Visitor Location Register), and MSC (Mobile Switching Center). Key terms such as ISDN, PSPDN, and GSPDN are also mentioned.

Legend:
- SIM: Subscriber Identity Module
- ME: Mobile Equipment
- BTS: Base Transceiver Station
- BSC: Base Station Controller
- HLR: Home Location Register
- VLR: Visitor Location Register
- MSC: Mobile Switching Center
- PSTN: Public Switched Telephone Network
- ISDN: Integrated Services Digital Network
- PSPDN: Public Switched Packet Data Network
- GSPDN: Gateway Switched Public Data Network
Processing library related to Telephony

call()

**Syntax**: p.call(number)

**Description**
Calls the specified phone number. Returns true if the sketch must terminate before the call will be initiated.

**Parameters**
- p Phone: any variable of type Phone
- number String: phone number to dial
Message (SMS)

SMSC : Short Message Service Centre

SMS Data Dissected

message HALLO WORLD
sent from the number ++39 347 3820955
at 04h:55m:16s PM of the 13th of January 2002.

- SMSC – Phone data protocol : specific protocols for GSM MAPf framework such as SS7 or use standard email protocol such as SMTP or TCP/IP
FlashLite library related to SMS

_capSMS

**Availability**: Flash Lite 1.1.

**Description**
Numeric variable; indicates whether Flash Lite can send *Short Message Service* (SMS) messages by using the `GetURL()` ActionScript command. If so, this variable is defined and has a value of 1; if not, this variable is undefined.

**Example**
The following example sets `canSMS` to 1 in Flash Lite 1.1, but leaves it undefined in Flash Lite 1.0 Flash Lite 1.0 (however, not all Flash Lite 1.1 phones can send SMS messages, so this code is still dependent on the phone):

**Code**
```javascript
on(release) {
    canSMS = _capSMS;
    if (canSMS) {
        // send an SMS
        myMessage = "sms:4156095555?body=sample sms message";
        getURL(myMessage);
    }
}
```
Processing library related to SMS

**EVENT_MSG_RECEIVED**

**Syntax:** Messenger.EVENT_MSG_RECEIVED

**Description**
This constant value is reported in `libraryEvent` when a message has been received. The data object is the `Message` received.

**Code**

```java
Messenger m;

void setup() {
    m = new Messenger(this); noLoop();
}

void libraryEvent(Object library, int event, Object data) {
    if (library == m) {
        switch (event) {
        case Messenger.EVENT_MSG_RECEIVED:
            Message msg = (Message) data;
            println(msg.readString());
            break;
        }
    }
}
```
M2M via SMS

- Control Motor device via GSM / GPRS wireless network

M2M related paper
Camera

- CCD / CMOS
- 5M pixel camera phone

- Moblog (Mobile + Weblog)
  - consists of content posted to the Internet from a mobile or portable device
generally involve technology which allows publishing from a mobile device

http://joi.ito.com/moblog/
Browser

- Microbrowser : web browser for handheld device considering low memory capacity
get contents written in WAP
Limited contents, service, needed WAP based page development

What is WAP?

- Mobile full browsing : web-like browser, Nokia open source browser
Mobile internet service is big issue
Browser

1. Opera browser
   Don’t need horizontal scroll

2. Nokia New Web Browser for S60
   Safari based open source browser

3. Openwave browser

4. Access Netfront browser

5. Obigo Teleca browser

6. Safari browser : iPhone
WAP (Wireless Application Protocol) is a major breakthrough that achieves universal Internet-based information access on wireless devices. It will make it possible for developers to write once for all networks worldwide. Carriers will be able to implement gateways that work with many brands of phones and all applications and content. Handset manufactures can make high volume, low cost handsets for all carriers. - WAP (Wireless Application Protocol), Shunxing Chen & Linfeng Yang

Bluetooth

- Connect 2 or more devices which are in proximity
- Don't require high bandwidth
- More flexible than IrDA
Bluetooth project using processing

Bluetooth library for processing

So here’s “RoombaCtrl”, a small Java program for your Bluetooth- and J2ME-compatible phone that works with the build-your-own Bluetooth adapter shown in the book “Hacking Roomba” or the pre-built RooTooth.

Useful Links for developer

- Do-It-Yourself MIDP on Mac OS X
- MPower free MIDP SDK for Mac OS X (the tutorial in the SDK is great)
- Avetana Bluetooth implementation
- J2ME Javadocs
- Bluetooth library source code from Mobile Processing
- Java APIs for Bluetooth
GPS (Global Positioning System)

- Satellite navigation system
  > more about GPS
Experiments using mobile devices

- RFID PAC-LAN Game

Four other players take the role of the “Ghosts” who attempt to hunt down the PAC-LAN player. A Java 2 Platform Micro Edition (J2ME) application, running on a mobile phone, is connected to a central server using a General Packet Radio Service (GPRS) connection. The server relays to the PAC-LAN character his points and the position of the game.
Experiments using mobile devices

- Location Based Game project 1

We describe two games in which online participants collaborated with mobile participants on the city streets. In the first, the players were online and professional performers were on the streets. The second reversed this relationship. Analysis of these experiences yields new insights into the nature of context. We show how context is more socially than technically constructed. We show how players exploited (and resolved conflicts between) multiple indicators of context including GPS, GPS error, audio talk, ambient audio, timing, local knowledge and trust. We recommend not overly relying on GPS, extensively using audio, and extending interfaces to represent GPS error.

![Image](image_url)
Experiments using mobile devices

- CELL PHONE: Art and Mobile Phone

Art and the Mobile Phone explores some of the groundbreaking works that are being created by artists today using cell phone technologies. These works engage such features and technologies as camera phones, video phones, global positioning systems, Bluetooth technology, ring tone sounds, and messaging. Artistic interest in mobile phone technology lies not only in producing artworks for individual handheld devices, but in the potential of mobile phone technologies to create works that can be performative and participatory.
Experiments using mobile devices

- WIFFITI

Wiffiti is a combination of text messaging and graffiti (wireless graffiti) which The Economist credits being started in Europe. Wiffiti allows a passerby to text a message to be displayed on a screen, wall, television, you name it. Wiffiti is available in 8 locations currently, but is expected to grow. We expect most growth in wiffiti will be due to advertiser interest, but time will tell.
Experiments using mobile devices

- Textual Healing

'Fear Fighter' is an interactive street projection. Asking the question, "What are you afraid of?" You text message your fears to the displayed number and they appear in his thoughts as he guns them down.
Experiments using mobile devices

- Colour by numbers

1. Call +46 (0) 70 57 57 807 and follow the instructions to colour the tower. The call has a time limit of 5 minutes, so that as many people as possible can get through.
2. Select the floors you wish to colour using the **buttons 0 to 9** on your phone. 0 is the topmost floor, and 9 the bottommost. Finish by pressing #. For example, if you want to colour the topmost four floors, you press 0123#. The colours on these floors are now controlled at the same time.
3. Colour the selected floors. Use the number buttons on your phone to mix red, green and blue until you get the colour you want.