Developing for Mobile Devices
Mobile Phones

• In 1980 McKinsey predicted around 1 million subscribers worldwide in the year 2000
• Actual numbers - 1 million phones sold a day in 2000
• In 2005, 832 million sold, 2.28 million a day
• In 2006, 1.019 billion sold, 2.79 million a day
• Currently, the industry is moving away from pure PDAs, and instead phones or PDA/phone hybrids are becoming more popular
## 2006 Phone Sales

<table>
<thead>
<tr>
<th>Vendor</th>
<th>Sold (000s)</th>
<th>Percentage</th>
</tr>
</thead>
<tbody>
<tr>
<td>Nokia</td>
<td>347,500</td>
<td>34.1%</td>
</tr>
<tr>
<td>Motorola</td>
<td>217,400</td>
<td>21.3%</td>
</tr>
<tr>
<td>Samsung</td>
<td>118,000</td>
<td>11.6%</td>
</tr>
<tr>
<td>Sony Ericsson</td>
<td>74,800</td>
<td>7.3%</td>
</tr>
<tr>
<td>LG Electronics</td>
<td>64,400</td>
<td>6.3%</td>
</tr>
<tr>
<td>Others</td>
<td>197,800</td>
<td>19.4%</td>
</tr>
<tr>
<td><strong>TOTAL</strong></td>
<td><strong>1,019,900</strong></td>
<td><strong>100%</strong></td>
</tr>
</tbody>
</table>
Mobile Phones

• Good development environments
• Most run Java
• C# now primarily used on Windows Mobile phones
• C++ used on Symbian phones
• Java used for most small apps, like games
Common Functionality

• WAP/GPRS - mobile Internet
• Bluetooth
• 802.11 WiFi
• Accelerometers
• SMS (Short Message Service) - accidental success
• MMS (Multimedia Message Service)
• LBS (Location Based Services)
• Mobile games
• Mobile email - really took off with Blackberrys
## Mobile Phone Use (UK)

<table>
<thead>
<tr>
<th>Activity</th>
<th>(000s)</th>
<th>Percent</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sent text message</td>
<td>36,240</td>
<td>84.3%</td>
</tr>
<tr>
<td>Used photo messaging</td>
<td>12,877</td>
<td>29.9%</td>
</tr>
<tr>
<td>Browsed news and information</td>
<td>6,229</td>
<td>14.5%</td>
</tr>
<tr>
<td>Used personal email</td>
<td>2,721</td>
<td>6.3%</td>
</tr>
<tr>
<td>Purchased ringtone</td>
<td>2,343</td>
<td>5.4%</td>
</tr>
<tr>
<td>Downloaded mobile game</td>
<td>1,737</td>
<td>4.0%</td>
</tr>
<tr>
<td>Used mobile IM</td>
<td>1,585</td>
<td>3.7%</td>
</tr>
<tr>
<td>Used work email</td>
<td>1,298</td>
<td>3.0%</td>
</tr>
<tr>
<td>Purchased wallpaper or screensaver</td>
<td>945</td>
<td>2.2%</td>
</tr>
</tbody>
</table>
## Different cultures

<table>
<thead>
<tr>
<th>UK: Activity</th>
<th>(000s)</th>
<th>Percent</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sent text message</td>
<td>36,240</td>
<td>84.3%</td>
</tr>
<tr>
<td>Used photo messaging</td>
<td>12,877</td>
<td>29.9%</td>
</tr>
<tr>
<td>Browsed news and information</td>
<td>6,229</td>
<td>14.5%</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>US: Activity</th>
<th>(000s)</th>
<th>Percent</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sent text message</td>
<td>70,864</td>
<td>37.3%</td>
</tr>
<tr>
<td>Used photo messaging</td>
<td>26,070</td>
<td>13.7%</td>
</tr>
<tr>
<td>Browsed news and information</td>
<td>20,709</td>
<td>10.9%</td>
</tr>
</tbody>
</table>
Java for mobiles

- Java Mobile Edition (Java ME)
  - A special version of Java for mobile devices
  - A cut-down set of classes you can use
- Two configurations for mobile and embedded devices
  - CDC: Connected Device Configuration
    - everything from pages to set-top boxes
    - almost all the Java framework classes, except for GUI
  - CLDC: Connected Limited Device Configuration
    - phones, PDAs
    - extremely minimal subset of the Java framework
Java for phones

- CLDC providing minimal, lightweight Java functionality
- **MIDP**: Mobile Information Device Profile
  - contains classes for IO, graphical design, and base midlet classes
- WMA: Wireless Messaging API
  - mobile communications
- MMAPI: Mobile Media API
  - classes for playing audio and video
- **MIDlet**: an application written to run on MIDP
Java for embedded devices

- CDC
- Foundation profile: almost everything other than GUI classes
- Personal Basis Profile: lightweight GUI support
- Personal Profile: extends PBP with full AWT support
- Developing for CDC is very similar to developing normal Java apps
Java ME

3D  Event  Location  Instant  Bluetooth
Web  Security  PDA specific  And more...
WMA  MMAPI

MIDP
CLDC

Phone or PDA

Personal Profile
Personal Basis Profile
Foundation Profile
CDC

Embedded system
MIDlets

- A standard Java application but...
- can only use classes in the MIDP/CLDC specification
- must define certain methods - startApp, pauseApp, destroyApp
package com.j2me.part1;
import java.util.Date;
import javax.microedition.lcdui.Alert;
import javax.microedition.lcdui.Display;
import javax.microedition.midlet.MIDlet;

public class DateTimeApp extends MIDlet {
    Alert timeAlert;

    public DateTimeApp() {
        timeAlert = new Alert("Alert!");
        timeAlert.setString(new Date().toString());
    }

    public void startApp() {
        Display.getDisplay(this).setCurrent(timeAlert);
    }

    public void pauseApp() {
        }

    public void destroyApp(boolean unconditional) {
        }
}

Example MIDlet
Mobile Phone Games

- Most are MIDlets
- Use javax.microedition.lcdui for graphics
- Allows development of games using Sprite and TiledLayer classes
Windows Mobile

• All new mobile devices that run Windows now run a version of Windows Mobile (6 is latest)

• WM is, unsurprisingly, very similar to desktop versions of Windows

• but much reduced functionality

• for example, only basic support for

• serialization

• accessing web services
Visual Studio

• First released in 1997

• **IDE** - Integrated Development Environment

• Attempt to use the same IDE to support multiple languages (Visual Basic, C++ and Java at the start)

• Initially, just a common design environment

• Current version is Visual Studio 2005
  • Upcoming version is Visual Studio 2008

• Free ‘Express’ versions for developing in a single language

• msdn.microsoft.com/vstudio/express
Most Used IDEs

- Visual Studio - 31.7%
- Adobe Studio - 11.8%
- Eclipse - 11.2%
.NET Framework

• MS hopes most applications developed now will use .NET

• Code is compiled to MSIL (Microsoft Intermediate Language), also frequently called CIL (Common Intermediate Language)

• Source can be compiled directly to native binary using the NGEN command, but this is rarely done
.NET runtime

- **CLR: Common Language Runtime**
- All .NET code compiles to the same CIL, regardless of the language used (C#, C++, Visual Basic, J#)
- CIL is JIT (Just In Time) compiled at runtime. The binary code can be cached by the .NET runtime, even between executions
C#  
C++  
Visual  
J#

Compile

MSIL/CIL

JIT (to binary)

Common Language Runtime

Store

.NET cache (optional, rare on mobiles)
Managed code

- .NET code is often called **managed code**
- It is executed and managed by the CLR
- Garbage collection / memory management is handled automatically
- Security is managed
  - for example, access to the file system or hardware can be restricted if the user desires
.NET advantages

• Can write code in any of the four .NET languages
• Can mix & match code
  • for example, one programmer can use C# for half an application and another can use J# for their half
• In theory, can run on multiple operating systems
.NET advantages

• Can use any of the languages to write ASP (Active Server Pages) to create scripted websites

• .NET runtime can provide extra level of security, as it is similar to a virtual machine

• To the user a .NET application looks exactly like a native Windows application, both on desktop and mobile
Mono

- Unsurprisingly, MS only made a CLR for Windows
- **Mono** is a set of tools to run .NET languages on Linux or OS X
  - compiler
  - CLR
- Is quite advanced, supporting low-level IO access and higher level GUI classes
C# (pronounced “See Sharp”) is a simple, modern, object-oriented, and type-safe programming language. C# has its roots in the C family of languages and will be immediately familiar to C, C++, and Java programmers.
C#

- MS recommends C# for all .NET development (desktop and mobile)
- C# does support slightly more functionality than other .NET languages
- For example, it can be much faster by using **unsafe code**
  - code that is not managed, allowing for pointers to be used and manual memory management
- C# developers earn 26% more than Visual Basic developers on average
- Currently the fastest growing language
- **REALLY** similar to Java
using System;

class HelloClass {
    public static void Main() {
        Console.WriteLine("Hello World");
    }
}
using System;
using System.Drawing;
using System.Windows.Forms;

namespace HelloWindowsFormsNamespaace {

    public class HelloForm : Form
    {
        // the label that will display the message
        private Label helloLabel;

        public HelloForm()
        {
            // set up label...
            helloLabel = new Label();
            helloLabel.Location = new Point(10, 10);
            helloLabel.Text = "Hello Windows Forms!";
            helloLabel.Size = new Size(400, 50);

            // add the label to the form's controls
            Controls.Add(helloLabel);

            // set the text of the form itself
            this.Text = "Hello World";
        }

        // This main function simply starts a new instance of our form
        public static void Main(string[] args)
        {
            Application.Run(new HelloForm());
        }
    }
}
DLLs

- Dynamic Link Library
- Commonly used throughout Windows
- Different programs can use the same DLL
  - for example, MS Word and Adobe Photoshop probably use the same Windows DLLs to read and write files
- So developers do not need to write code that does common functions, or include it in their program or installation CDs (so smaller programs)
A single copy of `kernel32.dll` is held in `c:\windows\system32\kernel32.dll`. Many programs will access this `kernel32` contains code for a method called `CreateFile`, which can create or open a stream of data, such as a file on a hard disk.
Accessing Native Code

- C# has excellent support for accessing native code

- System.InteropServices
  - A library providing Interprocess communication

- DllImport
  - A call allowing access to methods (code) held in any DLL (both managed and native)
Native access example

using System;
using System.Text;
using System.Runtime.InteropServices;

public class DllExample {

    [DllImport("advapi32.dll")]
    public static extern bool GetUserName(StringBuilder lpBuffer, ref int nSize);

    public static void Main(string args[]) {
        StringBuilder sb = new StringBuilder(100);
        GetUserName(sb, ref sb.Capacity);
        Console.WriteLine("Hello there "+ sb);
    }
}
Mobile native code

• Accessing native code is extremely important when using mobile devices
• Virtually all hardware drivers are written in native code
• Allows access to...
  • low-level operating system functions
    • reading battery levels, turning power off, vibrating a phone
  • internal and external hardware, such as sensors or communication devices
  • WiFi scanning, GPS units, accelerometers
Mobile native code

- Native code can also be much faster
- Great for small but repetitive tasks
  - Reading GPS data from a serial port
  - Reading accelerometer data as users walk around
  - Reading/writing to file
- How to use...
  - Write a native DLL using C++
  - Use System.InteropServices and DllImport to access your C++ DLL from .NET code
.NET and native code

• .NET can handle a lot of this for you

• When you call functions in the .NET library it, in turn, often calls native code

• For example, if you open a file with a .NET StreamReader it will actually call CreateFile in kernel32.dll... although the developer doesn’t have to see/know this

• When you use .NET Graphics to draw it actually calls native drawing methods, which are very fast

• But for non-standard/uncommon stuff you do need to make the native calls yourself
Summary

• Understanding mobile phone uses

• Developing for mobiles
  • knowing how to use VS is very important

• Next two HCI4 lessons are labs in the Boyd Orr, so meet there (in the smaller side room) on Thursday and Monday

• Lab 1: basic Visual Studio use and basic mobile development

• Lab 2: using DLLs, native code, accessing 802.11 and Bluetooth hardware

• Reminder: email final exercise ideas tomorrow (1st Nov)