**ICAPRG409A Develop Mobile Applications (Android using Eclipse)**

**Prerequisite**

If you choose the Android ( java ) language for this unit it is necessary to complete the unit on **ICAPRG414A Apply introductory programming skills in another language** before you can attempt this unit as the java programming skills and the Eclipse Integrated Development Environment (IDE) are covered in that unit.

**Resources**

Setup Eclipse with Android SDK (Software Development Kit) <http://developer.android.com/sdk/index.html> . This particular installation with the Android SDK allows for programming applications for Android mobile devices as well.

Eclipse WindowBuilder is a powerful and easy to use bi-directional Java GUI designer which saves time when coding the User Interface with its drag and drop visual WYSIWYG designer. <https://www.eclipse.org/windowbuilder/> is the home page of WindowBuilder with a full description of the proposal, design and the creation of the WindowBuilder Eclipse plugin.

**Android Tutorials**

1. Top 8 video tutorial sources

 <http://equallysimple.com>

2. <http://javatechig.com> - 40 plus videos Android Development

3. 200 video tutorials - outstanding

<https://www.youtube.com/playlist?list=PL2F07DBCDCC01493A>

Extensive tutorials are found at <http://developer.android.com/training/index.html>

**Set up Eclipse WindowBuilder with a GUI interface showing an ArrayList**.

<http://www.youtube.com/watch?v=b7ejd7gAynM>

**Set up Eclipse WindowBuilder with a multi-window GUI interface**

<http://www.youtube.com/watch?v=bKPGEqJHWaE>

**Additional resources**

<http://java67.blogspot.sg/2012/08/how-to-sort-arraylist-in-java-list.html>

<http://javarevisited.blogspot.com.au/2012/03/how-to-loop-arraylist-in-java-code.html>

<http://www.eclipseonetips.com/>

**Save data using XML serialization**

[http://robertmassaioli.wordpress.com/2011/04/21/simple-xml-in-android-1-5-and-up/](https://exchange.ringwoodsc.vic.edu.au/owa/redir.aspx?C=9a61119fde9c44eca2e672c500f6c382&URL=http%3a%2f%2frobertmassaioli.wordpress.com%2f2011%2f04%2f21%2fsimple-xml-in-android-1-5-and-up%2f)

**Create a Web Service and Consume it**

* [**https://www.youtube.com/watch?v=9QlJhQ-FfQ4**](https://www.youtube.com/watch?v=9QlJhQ-FfQ4)
* <https://www.youtube.com/watch?v=-3w6LBl8E-8>

**Consume a web service already available (weather)**

* <http://www.javacodegeeks.com/2013/06/android-build-real-weather-app-json-http-and-openweathermap.html>

Downloadable at <https://github.com/survivingwithandroid/Swa-app/archive/master.zip>

Get location for an address to pipe into weather service

* http://www.survivingwithandroid.com/2013/08/android-app-development-weather-app.html

**Java Videos**

Unless otherwise shown in a direct link the videos referenced in this document on general java programming can be found at one of 2 locations:

1. <http://www.youtube.com/user/mcgiggles373/videos>
2. <http://www.youtube.com/user/rithustutorials/videos>

**Android Videos**

|  |  |
| --- | --- |
| 1. Define a platform | 1.1 Determine target platform for the hand-held device1.2 Choose and evaluate a development environment based on the target platform, hardware and mobile phone manufacturer on which the application will run1.3 Prepare and configure the application development environment |
| 2. Design user interface for a mobile application | 2.1 Create views for different screen orientations and resolutions2.2 Integrate appropriate navigation techniques2.3 Design page layout and content2.4 Integrate appropriate user input techniques |
| 3. Permit the personalisation of parameters | 3.1 Write code to allow users to configure application settings3.2 Write code to allow users to configure application output, such as text size, brightness and sound effects |
| 4. Build user interface for a mobile application | 4.1 Itemise the required functionality4.2 Build user interface in the determined development environment to fulfil this functionality |
| 5. Connect to data sources | 5.1 Bind controls to data sources5.2 Persist data using eXtensible markup language (XML) serialisation5.3 Access remote data using XML web services |
| 6. Test and debug a mobile device application | 6.1 Test application for overall functionality according to requirements6.2 Iterate application design or build until test results meet requirements |
| 7. Deploy a mobile device application | 7.1 Create application packaging for deployment7.2 Deploy an application to target devices |

Required skills

* analytical skills to:
* design for small devices
* evaluate appropriate mobile development environment
* communication skills to provide advice and guidance to others
* literacy skills to read and interpret technical and non-technical information
* numeracy skills to make basic calculations for specifying the layout of the mobile user interface design
* problem-solving skills to:
* address common problems in building and deploying mobile applications
* perform basic programming debugging skills
* research skills to find and evaluate relevant technologies in mobile applications development
* technical skills to:
* create XML documents
* design user interface
* develop software applications
* test and debug applications
* use an integrated development environment (IDE).

Required knowledge

* basic knowledge of:
* hardware and networking
* object-oriented programming
* user-interface design
* web design: HTML, JavaScript, cascading style sheet (CSS), and AJAX
* detailed knowledge of XML programming and web services.

|  |  |
| --- | --- |
| Overview of assessment |  |
| Critical aspects for assessment and evidence required to demonstrate competency in this unit | Evidence of the ability to:* design, build, test and deploy applications for small devices:
* mobile phones
* pocket PCs
* personal digital assistants (PDAs)
* enterprise digital assistants (EDAs).
 |
| Context of and specific resources for assessment | Assessment must ensure access to:* hand-held devices or stimulators
* integrated development environment (IDE)
* internet and web services
* server system with sufficient privileges to deploy applications
* specific tools and licences, depending on particular platform
* appropriate learning and assessment support when required
* modified equipment for people with special needs.
 |
| Method of assessment | A range of assessment methods should be used to assess practical skills and knowledge. The following example is appropriate for this unit: * verbal or written questioning to assess knowledge of mobile development environment.
 |

**Introduction**

The skills and knowledge needed to demonstrate all the elements and performance criteria in this unit on Introductory programming skills in another language can be achieved in many ways.

One popular and powerful IDE (Integrated Development Environment) is Eclipse. The following is an outline of eclipse as taken from Wikipedia [http://en.wikipedia.org/wiki/Eclipse\_(software)](http://en.wikipedia.org/wiki/Eclipse_%28software%29)

In [computer programming](http://en.wikipedia.org/wiki/Computer_programming), **Eclipse** is an [integrated development environment](http://en.wikipedia.org/wiki/Integrated_development_environment) (IDE). It contains a base [workspace](http://en.wikipedia.org/wiki/Workspace) and an extensible [plug-in](http://en.wikipedia.org/wiki/Plug-in_%28computing%29) system for customizing the environment. Written mostly in [Java](http://en.wikipedia.org/wiki/Java_%28programming_language%29), Eclipse can be used to develop applications. By means of various plug-ins, Eclipse may also be used to develop applications in other [programming languages](http://en.wikipedia.org/wiki/Programming_language): [Ada](http://en.wikipedia.org/wiki/Ada_%28programming_language%29), [ABAP](http://en.wikipedia.org/wiki/ABAP), [C](http://en.wikipedia.org/wiki/C_%28programming_language%29), [C++](http://en.wikipedia.org/wiki/C%2B%2B), [COBOL](http://en.wikipedia.org/wiki/COBOL), [Fortran](http://en.wikipedia.org/wiki/Fortran), [Haskell](http://en.wikipedia.org/wiki/Haskell_%28programming_language%29), [JavaScript](http://en.wikipedia.org/wiki/JavaScript), [Lasso](http://en.wikipedia.org/wiki/Lasso_%28programming_language%29), [Perl](http://en.wikipedia.org/wiki/Perl), [PHP](http://en.wikipedia.org/wiki/PHP), [Python](http://en.wikipedia.org/wiki/Python_%28programming_language%29), [R](http://en.wikipedia.org/wiki/R_%28programming_language%29), [Ruby](http://en.wikipedia.org/wiki/Ruby_%28programming_language%29) (including [Ruby on Rails](http://en.wikipedia.org/wiki/Ruby_on_Rails) framework), [Scala](http://en.wikipedia.org/wiki/Scala_%28programming_language%29), [Clojure](http://en.wikipedia.org/wiki/Clojure), [Groovy](http://en.wikipedia.org/wiki/Groovy_%28programming_language%29), [Scheme](http://en.wikipedia.org/wiki/Scheme_%28programming_language%29), and [Erlang](http://en.wikipedia.org/wiki/Erlang_%28programming_language%29). It can also be used to develop packages for the software [Mathematica](http://en.wikipedia.org/wiki/Mathematica). Development environments include the Eclipse Java development tools (JDT) for Java and Scala, Eclipse CDT for C/C++ and Eclipse PDT for PHP, among others.

The initial [codebase](http://en.wikipedia.org/wiki/Codebase) originated from [IBM VisualAge](http://en.wikipedia.org/wiki/IBM_VisualAge).[[2]](http://en.wikipedia.org/wiki/Eclipse_%28software%29#cite_note-VisualAge-2) The Eclipse [software development kit](http://en.wikipedia.org/wiki/Software_development_kit) (SDK), which includes the Java development tools, is meant for Java developers. Users can extend its abilities by installing plug-ins written for the Eclipse Platform, such as development toolkits for other programming languages, and can write and contribute their own plug-in modules.

Released under the terms of the [Eclipse Public License](http://en.wikipedia.org/wiki/Eclipse_Public_License), Eclipse [SDK](http://en.wikipedia.org/wiki/Software_development_kit) is [free and open source software](http://en.wikipedia.org/wiki/Free_and_open_source_software) (although it is incompatible with the [GNU General Public License](http://en.wikipedia.org/wiki/GNU_General_Public_License)[[3]](http://en.wikipedia.org/wiki/Eclipse_%28software%29#cite_note-3)). It was one of the first IDEs to run under [GNU Classpath](http://en.wikipedia.org/wiki/GNU_Classpath) and it runs without problems under [IcedTea](http://en.wikipedia.org/wiki/IcedTea).

## History

Eclipse began as an [IBM Canada](http://en.wikipedia.org/wiki/IBM_Canada) project. [Object Technology International](http://en.wikipedia.org/wiki/Object_Technology_International) (OTI), which had previously marketed the [Smalltalk](http://en.wikipedia.org/wiki/Smalltalk)-based [VisualAge](http://en.wikipedia.org/wiki/VisualAge) family of [integrated development environment](http://en.wikipedia.org/wiki/Integrated_development_environment) (IDE) products,[[2]](http://en.wikipedia.org/wiki/Eclipse_%28software%29#cite_note-VisualAge-2) developed the new product as a Java-based replacement.[[4]](http://en.wikipedia.org/wiki/Eclipse_%28software%29#cite_note-4) In November 2001, a consortium was formed with a board of stewards to further the development of Eclipse as open-source software. It is estimated that IBM had already invested close to $40 million by that time.[[5]](http://en.wikipedia.org/wiki/Eclipse_%28software%29#cite_note-5) The original members were [Borland](http://en.wikipedia.org/wiki/Borland), [IBM](http://en.wikipedia.org/wiki/IBM), [Merant](http://en.wikipedia.org/wiki/Micro_Focus_International), [QNX Software Systems](http://en.wikipedia.org/wiki/QNX_Software_Systems), [Rational Software](http://en.wikipedia.org/wiki/Rational_Software), [Red Hat](http://en.wikipedia.org/wiki/Red_Hat), [SuSE](http://en.wikipedia.org/wiki/SuSE), [TogetherSoft](http://en.wikipedia.org/wiki/Borland_Together) and [WebGain](http://en.wikipedia.org/wiki/WebGain).[[6]](http://en.wikipedia.org/wiki/Eclipse_%28software%29#cite_note-6) The number of stewards increased to over 80 by the end of 2003. In January 2004, the [Eclipse Foundation](http://en.wikipedia.org/wiki/Eclipse_Foundation) was created.[[7]](http://en.wikipedia.org/wiki/Eclipse_%28software%29#cite_note-7)

Eclipse 3.0 (released on 21 June 2004) selected the [OSGi](http://en.wikipedia.org/wiki/OSGi) Service Platform specifications as the runtime architecture.[[8]](http://en.wikipedia.org/wiki/Eclipse_%28software%29#cite_note-8)

The [Association for Computing Machinery](http://en.wikipedia.org/wiki/Association_for_Computing_Machinery) recognized Eclipse with the 2011 [ACM Software Systems Award](http://en.wikipedia.org/wiki/ACM_Software_Systems_Award) on 26 April 2012.[[9]](http://en.wikipedia.org/wiki/Eclipse_%28software%29#cite_note-9)

### Licensing

The [Eclipse Public License](http://en.wikipedia.org/wiki/Eclipse_Public_License) (EPL) is the fundamental license under which Eclipse projects are released.[[10]](http://en.wikipedia.org/wiki/Eclipse_%28software%29#cite_note-10) Some projects require dual licensing, for which the Eclipse Distribution License (EDL) is available, although use of this license must be applied for and is considered on a case-by-case basis.

Eclipse was originally released under the [Common Public License](http://en.wikipedia.org/wiki/Common_Public_License), but was later relicensed under the Eclipse Public License. The [Free Software Foundation](http://en.wikipedia.org/wiki/Free_Software_Foundation) has said that both licenses are [free software](http://en.wikipedia.org/wiki/Free_software) licenses, but are incompatible with the [GNU General Public License](http://en.wikipedia.org/wiki/GNU_General_Public_License) (GPL).[[11]](http://en.wikipedia.org/wiki/Eclipse_%28software%29#cite_note-11) [Mike Milinkovich](http://en.wikipedia.org/wiki/Mike_Milinkovich), of the Eclipse Foundation commented that moving to the GPL would be considered when version 3 of the GPL was released.[[12]](http://en.wikipedia.org/wiki/Eclipse_%28software%29#cite_note-12)

### Name

According to Lee Nackman, [Chief Technology Officer](http://en.wikipedia.org/wiki/Chief_Technology_Officer) of IBM's [Rational division](http://en.wikipedia.org/wiki/Rational_Software) (originating in 2003) at that time, the name "Eclipse" (dating from at least 2001) was not a wordplay on [Sun Microsystems](http://en.wikipedia.org/wiki/Sun_Microsystems), as the product's primary competition at the time of naming was [Microsoft Visual Studio](http://en.wikipedia.org/wiki/Microsoft_Visual_Studio) (which it, *Eclipse*, was to eclipse).[[13]](http://en.wikipedia.org/wiki/Eclipse_%28software%29#cite_note-13)

### Architecture

Eclipse uses plug-ins to provide all the functionality within and on top of the runtime system. Its runtime system is based on [Equinox](http://en.wikipedia.org/wiki/Equinox_%28OSGi%29), an implementation of the [OSGi](http://en.wikipedia.org/wiki/OSGi) core framework specification.[*[citation needed](http://en.wikipedia.org/wiki/Wikipedia%3ACitation_needed%22%20%5Co%20%22Wikipedia%3ACitation%20needed)*]

In addition to allowing the Eclipse Platform to be extended using other [programming languages](http://en.wikipedia.org/wiki/Programming_language) such as [C](http://en.wikipedia.org/wiki/C_%28programming_language%29) and [Python](http://en.wikipedia.org/wiki/Python_%28programming_language%29), the plug-in framework allows the Eclipse Platform to work with typesetting languages like [LaTeX](http://en.wikipedia.org/wiki/LaTeX),[[28]](http://en.wikipedia.org/wiki/Eclipse_%28software%29#cite_note-29) networking applications such as [telnet](http://en.wikipedia.org/wiki/Telnet) and [database management systems](http://en.wikipedia.org/wiki/Database_management_system). The plug-in architecture supports writing any desired extension to the environment, such as for [configuration management](http://en.wikipedia.org/wiki/Configuration_management). Java and [CVS](http://en.wikipedia.org/wiki/Concurrent_Versions_System) support is provided in the Eclipse [SDK](http://en.wikipedia.org/wiki/Software_development_kit), with support for other [version control systems](http://en.wikipedia.org/wiki/Version_control_system) provided by third-party plug-ins.

With the exception of a small run-time kernel, everything in Eclipse is a plug-in. This means that every plug-in developed integrates with Eclipse in exactly the same way as other plug-ins; in this respect, all features are "created equal".[[*citation needed*](http://en.wikipedia.org/wiki/Wikipedia%3ACitation_needed)] Eclipse provides plug-ins for a wide variety of features, some of which are through third parties using both free and commercial models. Examples of plug-ins include for [UML](http://en.wikipedia.org/wiki/Unified_Modeling_Language), for Sequence and other UML diagrams, a plug-in for DB Explorer, and many others.

The Eclipse SDK includes the Eclipse Java development tools (JDT), offering an IDE with a built-in [incremental](http://en.wikipedia.org/wiki/Incremental_compiler) Java compiler and a full model of the Java source files. This allows for advanced [refactoring](http://en.wikipedia.org/wiki/Refactor) techniques and code analysis. The IDE also makes use of a *workspace*, in this case a set of [metadata](http://en.wikipedia.org/wiki/Metadata) over a flat filespace allowing external file modifications as long as the corresponding workspace "resource" is refreshed afterwards.

Eclipse implements [widgets](http://en.wikipedia.org/wiki/GUI_widget) through a Java toolkit called [SWT](http://en.wikipedia.org/wiki/Standard_Widget_Toolkit), whereas most Java applications use the Java standard [Abstract Window Toolkit](http://en.wikipedia.org/wiki/Abstract_Window_Toolkit) (AWT) or [Swing](http://en.wikipedia.org/wiki/Swing_%28Java%29). Eclipse's user interface also uses an intermediate [graphical user interface](http://en.wikipedia.org/wiki/Graphical_user_interface) layer called [JFace](http://en.wikipedia.org/wiki/JFace), which simplifies the construction of applications based on SWT.

Language packs being developed by the "Babel project" provide translations into over a dozen [natural languages](http://en.wikipedia.org/wiki/Natural_language).[[29]](http://en.wikipedia.org/wiki/Eclipse_%28software%29#cite_note-30)

### Installation

### Eclipse and Android SDK

### Firstly download the Java JDK for your system from <http://www.oracle.com/us/downloads/index.html>

### Setup Eclipse with Android SDK (Software Development Kit) by downloading the 32bit or the 64 bit version for Windows by confirming you’re your instructor first followed by the installation <http://developer.android.com/sdk/index.html>

### Eclipse and WindowBuilder

### Setup WindowBuilder <https://www.eclipse.org/windowbuilder/> guided by your instructor and ensure it is working as expected.

### Running Eclipse

### To run Eclipse browse to the folder with find the executable file eclipse.exe. A short cut can be created for this file on the desktop to allow an easy startup link. Once the Workspace is confirmed by clicking OK, the Eclipse environment will appear as follows:

###

###

### Environment

### After the setup of Eclipse with the Android SDK and the WindowBuilder plugin it is time to become familiar with the environment. This will be covered by a combination of videos, notes and exercises and activities.

1. **Define a platform**

1.1 Determine target platform for the hand-held device

1.2 Choose and evaluate a development environment based on the target platform, hardware and mobile phone manufacturer on which the application will run

1.3 Prepare and configure the application development environment

### The target platform for the hand-held device will be Java ME ( Micro Edition ) and the development environment is to be the Eclipse IDE with the Android SDK ( Systems Development Kit ) and the WindowBuilder plug in.

### Java ME, short for Java Platform, Micro Edition, is a Java platform, used for embedded systems. Mobile phones, especially [feature phones](http://mobiledevices.about.com/od/glossary/g/What-Is-A-Feature-Phone.htm), most use this platform. Java ME was earlier referred to as J2ME.

### The Java ME source code is now licensed under the GNU General Public License. At present, all Java ME platforms are limited to JRE 1.3 features and employ the very same version of the class file format.

### *Activity 1.1*

### Complete the Building your First App activity found at <http://developer.android.com/training/index.html>

**2. Design a user interface for a mobile application**

2.1 Create views for different screen orientations and resolutions

2.2 Integrate appropriate navigation techniques

2.3 Design page layout and content

2.4 Integrate appropriate user input techniques

### *Activity 2.1*

### Complete the Supporting Different Devices activity found at <http://developer.android.com/training/index.html>

### Complete the Adding the Action Bar activity found at <http://developer.android.com/training/index.html>

### Complete the Building a Dynamic UI with Fragments activity found at <http://developer.android.com/training/index.html>

### Complete the Managing the Activity LifeCycle activity found at <http://developer.android.com/training/index.html>

### *Activity 2.2*

### Under the section *Best Practices for User Input* at <http://developer.android.com/training/index.html> complete the following activities

### Using touch gestures

### Handling keyboard input

1. **Permit the personalisation of parameters**

3.1 Write code to allow users to configure application settings

3.2 Write code to allow users to configure application output, such as text size, brightness and sound effects

### *Activity 3.1*

### Under the section *Best Practices for Interaction and Engagement* at <http://developer.android.com/training/index.html> complete the following activities

### Designing effective navigation

### Implementing effective navigation

### Notifying the user

### Adding search functionality

### Making your app content searchable by google

**4. Build user interface for a mobile application**

4.1 Itemise the required functionality

4.2 Build user interface in the determined development environment to fulfil this functionality

### *Activity 4.1*

### Under the section *Best Practices for User Interface* at <http://developer.android.com/training/index.html> complete the following activities

### Designing for multiple screens

### Designing for TV

### Creating Custom Views

### Creating Backward compatible Uis

### Implementing accessibility

### Managing the System UI

### *Activity 4.2*

### Under the section *Designing for Multiple Screens* at <http://developer.android.com/training/index.html> complete the following activities

### Supporting different screen sizes

### Supporting different screen densities

### Implementing adaptive UI flows

**5. Connect to data sources**

5.1 Bind controls to data sources

5.2 Persist data using eXtensible markup language (XML) serialisation

5.3 Access remote data using XML web services

### *Activity 5.1*

### Under the section *Saving Data* at <http://developer.android.com/training/index.html> complete the following activities

### Saving Key-value sets

### Saving Files

### Saving Data in SQL databases

### *Activity 5.2*

### Under the section *User Info and Location* at <http://developer.android.com/training/index.html> complete the following activities

### Accessing Contacts Data

### Making your app location aware

### *Activity 5.3*

### Under the section *Content Sharing* at <http://developer.android.com/training/index.html> complete the following activities

### Sharing simple data

### *Activity 5.4*

### Under the section *Interacting with other apps* at <http://developer.android.com/training/index.html> complete the following activities

### Designing for multiple screens

### *Activity 5.5*

1. **Create a Web Service and Consume it**
	* [**https://www.youtube.com/watch?v=9QlJhQ-FfQ4**](https://www.youtube.com/watch?v=9QlJhQ-FfQ4)
	* <https://www.youtube.com/watch?v=-3w6LBl8E-8>
2. **Consume a web service already available (weather)**
* <http://www.javacodegeeks.com/2013/06/android-build-real-weather-app-json-http-and-openweathermap.html>

Downloadable at <https://github.com/survivingwithandroid/Swa-app/archive/master.zip>

Get location for an address to pipe into weather service

* <http://www.survivingwithandroid.com/2013/08/android-app-development-weather-app.html>

**Organisational Coding Standards**

When programs are written in an organization such as a school, a government department, a charity, a bank, an insurance company or a software producing company, there are standards or requirements as part of the writing of the software.

Coding standards involve the following:

* Coding standards are guidelines for code style and documentation.
* The dream is that any developer familiar with the guidelines can work on any code that followed them.
* Standards range from a simple series of statements to involved documents.

Coding standards cover:

* Program Design
* Naming Conventions
* Formatting Conventions
* Documentation
* Possibly Even Licensing

The reasons for having coding standards include:

* Greater consistency between developers
* Easier to develop and maintain
* Saves time and money

Reducing the cost of software maintenance is the most often cited reason for following coding conventions. In their introduction to code conventions for the Java programming language, Sun Microsystems provides the following rationale:[[1]](http://en.wikipedia.org/wiki/Coding_conventions%22%20%5Cl%20%22cite_note-1)

Code conventions are important to programmers for a number of reasons:

* 40%-80% of the lifetime cost of a piece of software goes to maintenance. [[2]](http://en.wikipedia.org/wiki/Coding_conventions#cite_note-2)
* Hardly any software is maintained for its whole life by the original author.
* Code conventions improve the readability of the software, allowing engineers to understand new code more quickly and thoroughly.
* If you ship your source code as a product, you need to make sure it is as well packaged and clean as any other product you create.

There is not much point in writing a program that is not easy to follow. A program can be made easier follow if there are a number of principles that all programmers pay attention to, such as:

* Use of consistent, meaningful variable names eg. ***wages*** for a worker’s wages, ***hoursWorked*** for the number of hours a worker has worked. Compare that with using x for a worker’s wages, and y for the number of hours worked. The former is very easy to read and follow, the latter is close to impossible to make sense of as only the original programmer would know what he/she means by x or y.
* Use of a naming convention. A number of naming conventions exist such as CamelCase. For some languages (such as [Java](http://en.wikipedia.org/wiki/Java_%28programming_language%29) and [Microsoft](http://en.wikipedia.org/wiki/Microsoft)'s [.NET](http://en.wikipedia.org/wiki/Microsoft_.NET)) this practice is recommended by the language developers or by authoritative manuals and has therefore become part of the language's "culture". Examples of CamelCase include :
	+ myHourlyRate
	+ hoursWorked
	+ sumOfHoursWorked
	+ totalTaxPayable

Note the first character is lower case, then each new word in the compound name is upper case.

* Style guidelines often distinguish between upper and lower camel case, typically specifying which variety should be used for specific kinds of entities: [variables](http://en.wikipedia.org/wiki/Variable_%28computer_science%29), [record fields](http://en.wikipedia.org/wiki/Field_%28computer_science%29), [methods](http://en.wikipedia.org/wiki/Method_%28computer_science%29), [procedures](http://en.wikipedia.org/wiki/Procedure_%28computer_science%29), [types](http://en.wikipedia.org/wiki/Type_%28computer_science%29), etc. These rules are sometimes supported by [static analysis](http://en.wikipedia.org/wiki/Static_code_analysis) tools that check source code for adherence. This means when a method name or variable name is entered the Integrated Development Environment (IDE) will produce a message or warning reminding the programmer of the correct style guideline if it is not being followed

**Internal Documentation**

Computer [software](http://en.wikipedia.org/wiki/Software) is said to have **Internal Documentation** if the notes on how and why various parts of code operate is included within the [source code](http://en.wikipedia.org/wiki/Source_code) as comments. It is often combined with meaningful [variable](http://en.wikipedia.org/wiki/Variable_%28programming%29) names with the intention of providing potential future programmers a means of understanding the workings of the code.

This contrasts with external [documentation](http://en.wikipedia.org/wiki/Documentation), where programmers keep their notes and explanations in a separate document.

Internal documentation has become increasingly popular as it cannot be lost, and any programmer working on the code is immediately made aware of its existence and has it readily available.

1. **Test and debug a mobile device application**

6.1 Test application for overall functionality according to requirements

6.2 Iterate application design or build until test results meet requirements

**Testing Programming Code**

Software testing can be stated as the process of validating and verifying that a computer program/application/product:

* meets the requirements that guided its design and development,
* works as expected,
* can be implemented with the same characteristics,
* and satisfies the needs of stakeholders.

A primary purpose of testing is to detect software failures so that defects may be discovered and corrected. Testing cannot establish that a product functions properly under all conditions but can only establish that it does not function properly under specific conditions.[[4]](http://en.wikipedia.org/wiki/Software_testing#cite_note-Kaner1-4) The scope of software testing often includes examination of code as well as execution of that code in various environments and conditions as well as examining the aspects of code: does it do what it is supposed to do and do what it needs to do. In the current culture of software development, a testing organization may be separate from the development team. There are various roles for testing team members. Information derived from software testing may be used to correct the process by which software is developed.[[5]](http://en.wikipedia.org/wiki/Software_testing#cite_note-kolawa-5)

Every software product has a target audience. For example, the audience for video game software is completely different from banking software. Therefore, when an organization develops or otherwise invests in a software product, it can assess whether the software product will be acceptable to its end users, its target audience, its purchasers and other stakeholders. **Software testing** is the process of attempting to make this assessment.

 **Testing Table**

|  |  |  |  |
| --- | --- | --- | --- |
| **Test** | **Expected Result** | **Actual Result (**A supporting screen shot of the results would be required.**)** | **Conclusion/Action** |
| hours = 15 | Program to process and calculate total wages  | Program processes and outputs total wages correctly.  | Positive integers for hours will work correctly in the program. |
| User input hours = “hello” | Program to report an input error and ask user to input hours again as a positive integer | Program crashes with Exception Error  | Correct code to check for type of data entered for variable hours and output a message if not an integer. Request user to re-enter hours. |
| Calculate wages | Wages should be the correct product of hours worked times hourly rate | Wages is not the correct product of hours worked times hourly rate | Correct the code that calculates wages |

**Designing Test Data**

It is very important to design comprehensive testing data to ensure the program will operate in all possible environments. Examples of test data to include are:

* **existence testing –** check that data is entered and there isn’t a blank or lack of data.
* **range testing –** check that the value is within an acceptable or expected range eg. day of month cannot be less than 1 or greater than 31.
* **type testing –** if an integer is expected check that a different data type is not entered such as String or Boolean.
* **Reasonableness testing –** Is the data reasonable? For example, hours worked for an employee has to be greater than or equal to zero. There is no point accepting a negative value and proceeding with a wages calculation as the results will be meaningless. Equally there is no point in accepting a value of 168 for hours worked as this would mean the worker did not sleep but worked for 7 days straight.
* **Efficient and well-designed test data –** Testing is time consuming and must be efficiently designed to maximise the effectiveness of the testing process. For example if the program works with a positive value for hours worked such as 20 hours, there is not much point testing 21, 22, 23, 24, 25 etc hours as these will not give more insight into the program unless there is a boundary to be checked such as more than 20 hours per week means the hourly rate may change and this will impact on the wages calculated. Otherwise, testing has to be related to the logic of the actual code written.

**Sample Testing Cycle**

* **Test planning**: [Test strategy](http://en.wikipedia.org/wiki/Test_strategy), [test plan](http://en.wikipedia.org/wiki/Test_plan), [testbed](http://en.wikipedia.org/wiki/Testbed) creation. Since many activities will be carried out during testing, a plan is needed.
* **Test development**: Test procedures, [test scenarios](http://en.wikipedia.org/wiki/Scenario_test), [test cases](http://en.wikipedia.org/wiki/Test_case), test datasets, test scripts to use in testing software.
* **Test execution**: Testers execute the software based on the plans and test documents then report any errors found to the development team.
* **Test reporting**: Once testing is completed, testers generate metrics and make final reports on their [test effort](http://en.wikipedia.org/wiki/Test_effort) and whether or not the software tested is ready for release.
* **Test result analysis**: Or Defect Analysis, is done by the development team usually along with the client, in order to decide what defects should be assigned, fixed, rejected (i.e. found software working properly) or deferred to be dealt with later.
* **Defect Retesting**: Once a defect has been dealt with by the development team, it is retested by the testing team. AKA [Resolution testing](http://en.wikipedia.org/w/index.php?title=Resolution_testing&action=edit&redlink=1).
* **Regression testing**: It is common to have a small test program built of a subset of tests, for each integration of new, modified, or fixed software, in order to ensure that the latest delivery has not ruined anything, and that the software product as a whole is still working correctly.
* **Test Closure**: Once the test meets the exit criteria, the activities such as capturing the key outputs, lessons learned, results, logs, documents related to the project are archived and used as a reference for future projects.

Testing is a specialization in the Information Technology profession and there are many additional aspects to testing such as User Acceptance Testing, Alpha Testing and so on which are beyond the scope of this Unit of Competence.

### *Activity 6.1*

### Under the section *Testing your Android Activity* at <http://developer.android.com/training/index.html> complete the following activities

### Setting up your test environment

### Creating and running a test case

### Testing UI components

### Creating Unit tests

### Creating Functional tests

### *Activity 6.2*

### For any two of your previously created apps complete testing of UI components

### For any two of your previously created apps complete testing of Unit testing

### For any two of your previously created apps complete functional testing

1. **Deploy a mobile device application**

7.1 Create application packaging for deployment

7.2 Deploy an application to target devices

### *Activity 7.1*

### Deploy your app to an emulator

### Deploy an app to a physical Android phone