BAB II

LANDASAN TEORI

2.1. J2ME (*Java 2 Micro Edition*)

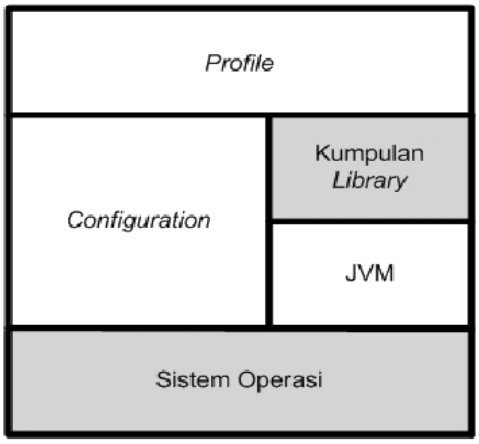
2.1.1. Pengertian J2ME

Java2 Micro Edition atau yang biasa disebut J2ME Menurut **Zuber Fraooqui, Vineet Richharia, Suneel Phulre (2011:113)** adalah *Java Platform, Micro Edition, or Java ME, is a Java platform designed for embedded systems (mobile devices are one kind of such systems*). Java merupakan bahasa pemograman berbasis obyek yang memiliki perkembangan yang pesat. Dengan keunggulan yang tidak dimiliki oleh bahasa pemrograman lainnya, yaitu multiplatform.

Sedangkan Menurut **M.Shalaluddin dan Rosa A.S (2008:3)** *Java2 Micro Edition* adalah lingkungan pengembangan yang didesain untuk meletakan perangkat lunak java pada barang elektronik beserta pendukungnya. Pada J2ME, jika perangkat lunak berfungsi baik pada sebuah perangkat maka belum tentu juga berfungsi baik pada perangkat lainnya.

J2ME mambawa java pada dunia informasi, komunikasi, dan perangkat komputansi selain perangkat komputer dan destop yang biasanya lebih kecil dari pada perangkat komputer destop. J2ME biasanya digunakan untuk telephone selular dan sejenisnya. J2ME adalah bagian dari J2SE, karena itu tidak semua *library* yang ada pada J2SE dapat digunakan pada J2ME. Tetapi J2ME

mempunyai beberapa *library* khusus yang tidak dimiliki J2SE. Arsitektur J2ME dapat dilihat pada gambar berikut :

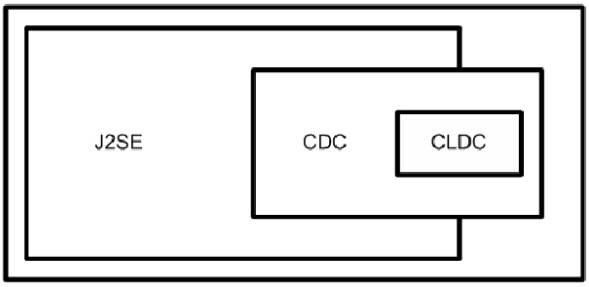


Gambar 2.1 Arsitektur J2ME
(Sumber : M. Shalaluddin dan Rosa A.S 2008, Pemrograman Java2 Micro Edition)

Teknologi J2ME juga memiliki beberapa keterbatasan, terutama jika diaplikasikan pada ponsel. J2ME sangat tergantung pada perangkat (*device*) yang digunakan, bisa dari segi merk ponsel, maupun kemampuan ponsel dan dukungannya terhadap teknologi J2ME. Misalnya, jika sebuah ponsel tidak memiliki kamera, maka jelas J2ME pada ponsel tersebut tidak dapat mengakses kamera. Keterbatasan lainnya adalah pada ukuran aplikasi, karena memori pada ponsel sangat terbatas. Sebagian besar ponsel tidak mengijinkan aplikasi J2ME menulis pada file karena alasan keamanan.

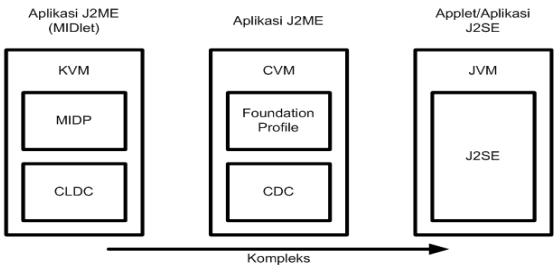
Configuration merupakan *Java library minimum* dan kapabilitas yang dipunya oleh para pengembang J2ME, yang maksudnya sebuah *mobile device* dengan kemampuan Java akan dioptimalkan untuk menjadi sesuai. *Configuration* hanya mengatur hal-hal tentang kesamaan sehingga dapat dijadikan ukuran kesesuaian antar *device*. Misalnya sebuah lampu sepeda dirancang sedemikian

sehingga dapat digunakan oleh berjenis-jenis sepeda. Dalam J2ME telah didefinisikan dua buah konfigurasi, yaitu CLDC (*Connected Limites Device Configuration*) untuk perangkat kecil dan CDC (*Connected Device Configuration*) untuk perangkat yang lebih besar. Lingkup CLDC dan CDC :



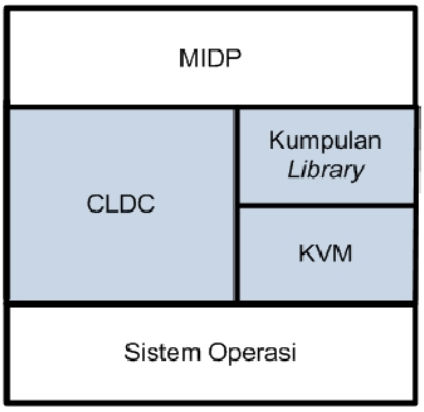
Gambar 2.2 Lingkup *Configuration*
(Sumber : M. Shalaluddin dan Rosa A.S 2008, Pemrograman Java2 Micro Edition)

Berbeda dengan *Configuration*, *Profile* membahas sesuatu yang spesifik untuk sebuah perangkat. Sebagai contoh misalnya, sebuah sepeda dengan merk tertentu, tentunya mempunyai ciri spesifik dengan sepeda lain. Dalam gambar J2ME terdapat dua buah *profile* yaitu MIDP dan *Foundation Profile.* Keterhubungan antara *configuration* dan *profile* yang ada pada J2ME beserta jenis mesin virtualnya dapat dilihat pada gambar berikut :



Gambar 2.3 Hubungan J2ME dengan J2SE
(Sumber : M. Shalaluddin dan Rosa A.S 2008, Pemrograman Java2 Micro Edition)

2.1.2 Connected Limited Device Configuration (CLDC)



Gambar 2.4 Arsitektur CLDC
(Sumber : M. Shalaluddin dan Rosa A.S 2008, Pemrograman Java2 Micro Edition)

CLDC atau Connected Limited Device Configuration adalah perangkat dasar dari J2ME, spesifikasi dasar berupa library dan API yangdiimplementasikan pada J2ME, seperti yang digunakan pada telepon seluler, pager, dan PDA. Perangkat tersebut dibatasi dengan keterbatasan memori, sumber daya, dan kemampuan memproses. Spesifikasi CLDC pada J2ME adalah spesifikasi minimal dari package, kelas, dan sebagian fungsi Java Virtual Machine yang dikurangi agar dapat diimplementasikan dengan keterbatasan sumber daya pada alat-alat tersebut, JVM yang digunakan disebut KVM (Kylobyte Virtual Machine).

2.1.3 Connected Device Configuration (CDC)

CDC atau Connected Device Configuration adalah super set dari CLDC. CDC menyediakan lingkungan Java runtime yang lebih luas dibandingkan CLDC dan lebih dekat kepada lingkungan J2SE. CDC terdiri dari virtual machine dan kumpulan library dasar untuk dipergunakan pada profile industri. Implementasi

CDC pada J2ME adalah source code yang menyediakan sambungan dengan macam-macam platform.

Perbedaan antara CDC dan CLDC dapat dilihat dari tabel dibawah ini :

Tabel 2.1 Perbandingnan CLDC dan CDC

CLDC	CDC
Mengimplementasikan sebagian dari J2SE.	Mengimplementasikan seluruh fitur J2SE.
JVM yang digunakan adalah KVM.	JVM yang digunakan adalah CVM.
Digunakan pada perangkat genggam (handphone, PDA, twoway pager) dengan memori terbatas 160-512 KB.	Digunakan pada perangkat genggam (internet TV, Nokia Communicator, car TV) dengan memori minimal 2M.
Prosesor : 16/32 bit.	Prosesor 32 bit.

2.2 Profil

Profil merupakan bagian perluasan dari konfigurasi, yaitu sekumpulan kelas yang terdapat pada konfigurasi, terdapat juga kelas-kelas spesifik yang didefinisikan lagi di dalam profil. Dengan kata lain, profil akan membantu secara fungsional yaitu dengan menyediakan kelas-kelas yang tidak terdapat di level konfigurasi. Berikut ini adalah profil J2ME yang tersedia saat ini yaitu :

1. MIDP (*Mobile Information Device Profile*)

   MIDP adalah profil yang disediakan oleh Sun Microsystems. MIDP menyediakan librari-librari Java untuk implementasi dasar antarmuka (GUI), implementasi jaringan (*networking*), database, dan timer. MIDP dirancang khususnya untuk *wireless phone* dan *pager.*

2. PDAP (*Personal Digital Assistant Profile*)

   Yaitu profil untuk PDA yang memperluas fungsi-fungsi pada konfigurasi CLDC dan digunakan khusus untuk menambahkan kemampuan-kemampuan lebih apabila dibandingkan dengan penggunaan profil MIDP.

3. *Foundation Profile*

   Yaitu profil yang digunakan untuk konfigurasi CDC. Profil ini menambahkan beberapa kelas dari J2SE ke dalam konfigurasi CDC, dan berperan juga sebagai pondasi untuk membentuk profil baru lainnya.

4. *Personal Profile*

   Yaitu profil yang mendefinisikan ulang PersonalJava sebagai profil yang dapat digunakan sebagai profil dalam J2ME. Profil ini merupakan hasil perluasan dari *Foundation Profile.*

5. *RMI Profile*

   Yaitu profil yang menambahkan dukungan RMI (*Remote Method Invocation*) ke dalam konfigurasi CDC.

2.2.1 MIDP atau Mobile Information Device Profile

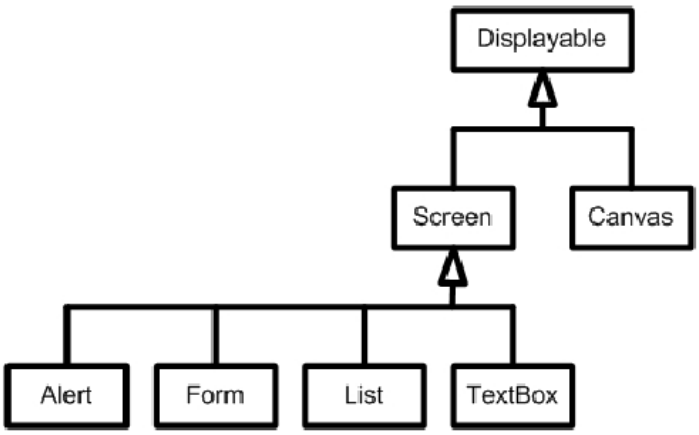
MIDP atau *Mobile Information Device Profile* adalah spesifikasi untuk sebuah profil J2ME. MIDP memiliki lapisan di atas CLDC, API tambahan untuk daur hidup aplikasi, antarmuka, jaringan dan penyimpanan persisten. Pada saat ini terdapat MIDP 1.0 dan MIDP 2.0. Fitur tambahan MIDP 2.0 dibanding MIDP 1.0 adalah API untuk multimedia. Pada MIDP 2,0 terdapat dukungan memainkan *tone, tone sequence* dan file WAV walaupun tanpa adanya *Mobile Media* API

(MMAPI). Posisi MIDP pada arsitektur J2ME dapat dilihat pada gambar di atas. Berikut adalah perbandingan MIDP 1.0 dan MIDP 2.0 :

Tabel 2.2 Perbandingan MIDP 1.0 dan MIDP 2.0

Spesifikasi	MIDP 1.0	MIDP 2.0
Display	96 x 54	96 x 54
Kedalaman display	1 -bit	1 -bit
Bentuk Piksel	Mendekati 1:1	Mendekati 1:1
Input	Keyboard dan touch screen	Keyboard dan touch screen
Memori	128 KB memori non-volatile untuk komponen MIDP. 8 KB memori non-volatile untuk data persistence yang dibuat oleh aplikasi. 32 KB memori volatile untuk JRE.	256 KB memori non-volatile untuk komponen MIDP. 8 KB memori non-volatile untuk data persistence yang dibuat oleh aplikasi. 128 KB memori volatile untuk JRE.
Jaringan	Dua arah, tanpa kabel (wireless)	Dua arah, tanpa kabel (wireless)
Library J2ME yang bukan merupakan library J2SE	javax.microedition.lcdui, javax.microedition.midlet, javax.microedition.rms	javax.microedition.lcdui, javax.microedition.midlet, javax.microedition.rms, javax.microedition.lcdui.game, javax.microedition.media, javax.microedition.pki
Multimedia		Memiliki kemampuan untuk memainkan file multimedia (suara dan video)

MIDP *User Interface* API memiliki API level tinggi dan level rendah. API level rendah berbasiskan penggunaan dari kelas abstrak *Canvas,* sedangkan kelas API level tinggi antara lain *Alert, Form, List* dan TextBox yang merupakan eksistensi dari kelas abstrak *Screen*. API level rendah lebih memberikan kemudahan kepada pengembang untuk memodifikasi sesuatu dengan kehendaknya, sedangkan API level tinggi biasanya hanya memberikan pengaksesan yang terbatas. Arsitektur antarmuka MIDP dapat dilihat pada gambar di bawah ini:



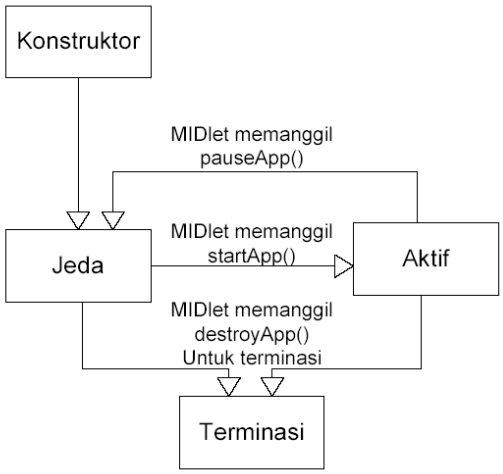
Gambar 2.5 MIDP User Interface
(Sumber : M. Shalaluddin dan Rosa A.S 2008, Pemrograman Java2 Micro Edition)

2.3 MIDlet

MIDlet adalah bagian dari kelas *javax.microedition.midlet.MIDlet* yang didefinisikan pada MIDP. MIDlet berupa sebuah kelas abstrak yang merupakan sub kelas dari bentuk dasar aplikasi sehingga antarmuka antara aplikasi J2ME dan aplikasi manajemen pada perangkat dapat terbentuk.

2.3.1 Daur hidup MIDlet

MIDlet terdiri dari beberapa metode yang harus ada, yaitu *construktor() protected void startApp() throws MIDletStateChangeException, protected void pauseApp(), protected void destroyApp(boolean unconditional) throws MIDletStateChangeException.* Alur hidup MIDlet dapat dilihat pada gambar 2.6.



Gambar 2.6 Alur hidup MIDlet
(Sumber : M. Shalaluddin dan Rosa A.S 2008, Pemrograman Java2 Micro Edition)

Ketika MIDlet dijalankan maka akan diinisialisasi dengan kondisi *pause* dan dijalankan *pauseApp(),* kondisi berikutnya adalah fungsi MIDlet dijalankan, yaitu pada *startApp().* Metode tersebut diimplementasikan sebagai *protected* dengan maksud agar MIDlet lain tidak dapat memanggil metode tersebut.

Ketika keluar dari MIDlet, maka metode *destroyApp()* akan dijalankan sebelum MIDlet benar-benar tidak berjalan lagi. *DestroyApp()* akan memanggil *notifyDestroyed(),* dan *notifyDestroyed()* akan memberitahu platform untuk menterminasi MIDlet dan membersihkan semua sumber daya yang mengacu pada MIDlet.

Dalam implementasinya, MIDlet memiliki struktur direktori antara lain sebagai berikut :

1. *Src*

   menyimpan *source code* untuk MIDlet dan kelas lain yang diperlukan.

2. *Res*

   Menyimpan sumber daya yang dibutuhkan oleh MIDlet, seperti misalnya gambar icon.

3. *Lib*

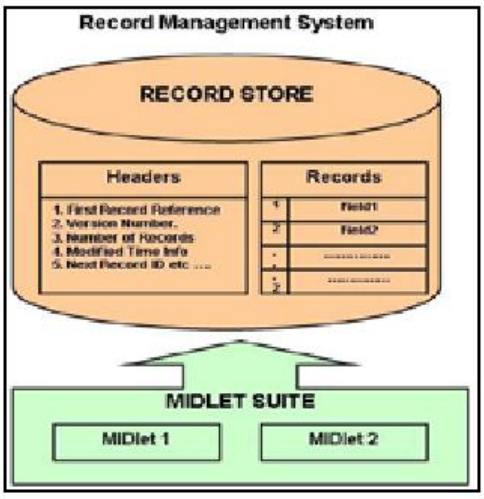
   Menyimpan file JAR atau ZIP yang berisi *library* tambahan yang dibutuhkan MIDlet.

4. *Bin*

   Menyimpan file JAR, JAD dan file *manifest* yang berisi muatan komponen MIDlet.

2.4 *Record Management System* (RMS)

Menurut **M.Shalahuddin dan Rosa A.S (2010 : 121)** *Record Management System* (RMS) adalah kumpulan record, dan record disimpan sebagai array dari byte dalam sebuah record store. RMS merupakan implementasi *database* sederhana untuk digunakan dalam MIDlet Java yang berorientasi *record*. Jika aplikasi kita ingin memanfaatkan fasilitas RMS ini, kita harus mengimport paket *Java*x.microedition.rms.*. Didalamnya terdapat *Class – Class* untuk manajemen RMS salah satunya *Class Record Store* yang menyediakan fungsi - fungsi untuk menamba, mengubah, menghapus data dan manipulasi lainnya.



Gambar 2.7 *Architecture of the RMS database*

(Sumber : Zuber Fraooqui Et Al. (2011) Improved Mechanism for Efficient object Pooling in J2ME Based Mobile Phone)

2.4.1. Paket *Javax*.microedition.rms

Paket ini menyediakan fungsi-fungsi untuk implementasikan *database* RMS pada MIDlet. Paket ini terdiri atas:

a. Interface

1. *RecordComparator*

   Mendefinisikan komparator atau pembanding ketika melakukan komparasi/perbandingan antara 2 *record*.

2. *RecordEnumerator*

   Merepresentasikan *enumerator record* yang bersifat *biderictional*(dua arah).

3. *RecordFilter*

   Mendefinisikan *filter* yang digunakan untuk menganalisis sebuah *record* pada *RecordStore*, apakah memenuhi kriteria tertentu sesuai dengan filter yang diberikan atau tidak.

4. *RecordListener*

   Mendeteksi *record* yang ditambahkan, diubah, atau dihapus dalam sebuah *RecordStore*.

b. Kelas

1. *RecordStore*

   Merupakan kelas utama yang harus Anda gunakan untuk membuat objek *RecordStore*.

c. Exception

1. *InvalidRecordIDException*

   Jika ditemukan *RecordId* yang tidak *valid*, eksepsi ini akan dilemparkan.

2. *RecordStoreException*

   Eksepsi umum yang digunakan jika terjadi kesalahan pada manajemen Objek *RecordStore*.

3. *RecordStoreFullException*

   Jika tempat penyimpanan *RecordStore* sudah penuh.

4. *RecordStoreNotFoundException*

   Jika tempat penyimpanan *RecordStore* tidak ditemukan.

5. *RecordStoreNotOpenException*

   Jika tempat penyimpanan *RecordStore* tidak dibuka (*opened*).

2.5 *Java Development Kit* (JDK)

Java Development Kit berisi perangkat lunak dan alat-alat yang Anda butuhkan untuk mengkompilasi, debug, dan menjalankan applet dan aplikasi bahwa Anda telah ditulis menggunakan bahasa pemrograman Java. JDK *(Java Development Kit)* sebagai komponen utama kumpulan alat pemrograman, termasuk Javac, jar, pengarsipan, yang paket perpustakaan kelas yang terkait ke dalam *file* JAR tunggal. Alat ini juga membantu mengelola *file* JAR, Javadoc - generator dokumentasi, yang secara otomatis menghasilkan dokumentasi dari komentar kode sumber, JDB - debugger, JPS - alat status proses, yang menampilkan informasi proses untuk proses Java saat ini, Javac yang disassembler *file* kelas , dan komponen lain begitu banyak.

JDK juga dilengkapi dengan *Java Runtime Environment* yang lengkap, biasanya disebut runtime pribadi. Ini terdiri dari *Java Virtual Machine* dan semua perpustakaan kelas hadir dalam lingkungan produksi, serta library tambahan hanya berguna untuk pengembang, dan seperti perpustakaan internasionalisasi dan perpustakaan IDL. Apa yang baru dalam versi ini: Versi 7 Update 2 berisi dukungan untuk Oracle Solaris 11 dan dukungan untuk Firefox 5 dan kemudian.

2.5.1. Komponen *Java Development Kit* (JDK)

Komponen *Java Development Kit* antara lain compiler Javac, interpreter Java disebut juga *Java virtual machine* atau *Java runtime environment*, applet viewer (*appletviewer*), debugger (jdb), *Java Class Library* (jcl), header dan stub generator (Javah), dan yang penting yaitu *Java documentation* (Javadoc).

Penjelasan penggunaan komponen JDK :

1. *Kompilator (*Javac*)*

   Berfungsi untuk kompilasi *file* source code : *.Java menjadi *.*Class* Syntax umum: Javac nama_*file*.*Java*.

2. *Interpreter (*Java*)*

   Bertugas untuk menjalankan bytecode (*.*Class*) Syntax umum :*Java* nama_*file*.*Class* .

3. *Applet Viewer*

   Digunakan untuk menjalanakan applet viewer , namun sekarang sudah digantikan browser. Syntax umum : appletviewer nama_*file*.html.

4. *Java Debugger*

   Bertugas untuk melakukan debugging aplikasi Java. Syntax umum : jdb option.

5. *Java Class File Diassembler* (*Java*p)

   Bertugas membuat daftar method dan attribute public dari suatu kelas Syntax Javap namaKelas.

6. *Java Header and Stub Generator*

   Bertugas menerjemahkan bahasa yang ditulis dalam bahasa Java menjadi bahasa pemrograman C. Syntax umum : *Java*h namaKelas

7. *Java Documentation Generator*

   Menampilkan pustaka kelas, interface, constructor, dan method standard yang telah dibuat vendor.Dari hasil instalasi JDK, dokumentasi ini dapat

dilihat pada C:\Java\docs\api\index.html dan Dari hasil instalasi NetBeans, dapat dilihat pada C:\Program *File*s\NetBeans3.6\doc\junit\index.html.

8. *Source Code Java API*

   Source code ini dapat diperoleh dari *file* src.zip. Untuk pemrogram pemula,lingkungan pemrograman Java dapat diringk menjadi :

   a. *Editing source code* menggunakan editor teks, seperti Notepad atau TextPad

   b. *Compiling* menggunakan *keyword* Javac melalui command prompt (dapat juga dari *editor teks* seperti TextPad)

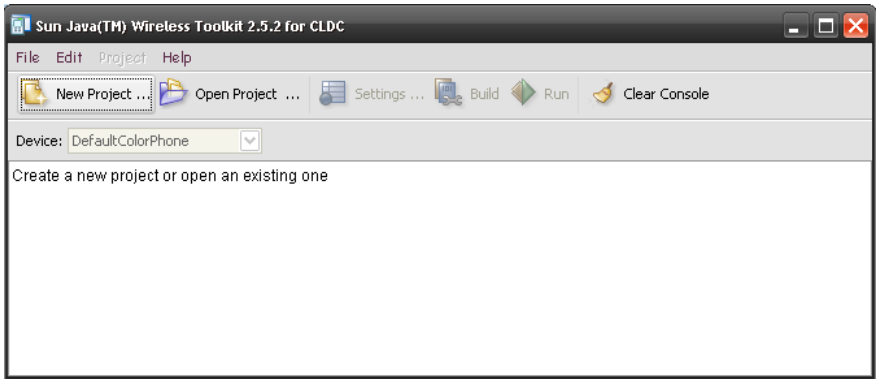
   c. *Executing* menggunakan :

      1. Command prompt untuk *Java application*(atau dari editor teks TextPad)

      2. *Browser* atau *appletviewer* untuk *Java* applet.

2.6. J2ME *Wireless Toolkit*

J2ME *Wirelesss Toolkit* adalah perkakas yang menyediakan lingkungan emulator, dokumentasi, beserta contoh-contoh aplikasi Java untuk perangkat kecil (small device). J2ME WTK berbasiskan pada CLDC dan MIDP. J2ME WTK adalah program yang meniru kerja ponsel yang mendukung MIDP atau yang biasa disebut emulator.



Gambar 2.8 Jendela *Wireless Toolkit*

(Sumber : M. Shalaluddin dan Rosa A.S 2008, Pemrograman Java2 Micro Edition)

2.6.1. *KToolbar*

KToolbar merupakan bagian utama dari aplikasi untuk pengembangan Java MIDP atau lebih dikenal dengan istilah MIDlet. *KToolbar* merupakan lingkunganpengembangan minimal yang disediakan oleh J2ME Wireless Toolkit untuk pengembangan MIDlet.Untuk melakukan pengembangan aplikasi MIDlet melaui *KToolbar* , hal pertama yang harus dilakukan adalah membuat proyek. Proyek ini secara fisik akandiletakkan pada C:\WTK2.5\apps\{nama-proyek}dengan asumsi J2ME Wireless Toolkit diinstal pada C:\WTK20\. Proyek yang dibuat terdiri atas direktori-direktori berikut:

a. Src , adalah direktori yang berisi berkas-berkas kode Java.
b. Bin, adalah direktori yang berisi berkas-berkas JAD, JAR dan manifest.
c. Res , adalah direktori yang berisi berkas-berkas sumber, misalnya gambar.
d. Lib, adalah direktori yang berisi berkas-berkas kelas Java yangdiperlukan dalam format gambar atau zip.

2.7. NetBeans

NetBeans mengacu pada dua hal, yakni Platform untuk pengembangan aplikasi desktop Java, dan sebuah Integrated Development Environment (IDE) yang dibangun menggunakan Platform NetBeans. Platform NetBeans memungkinkan aplikasi dibangun dari sekumpulan komponen perangkat lunak moduler yang disebut 'modul'. Sebuah modul adalah suatu arsip Java (*Java archive*) yang memuat kelas-kelas *Java* untuk berinetraksi dengan NetBeans Open API dan *file* manifestasi yang mengidentifikasinya sebagai modul. Aplikasi yang dibangun dengan modul-modul dapat dikembangkan dengan menambahkan modul-modul baru. Karena modul dapat dikembangkan secara independen, aplikasi berbasis *Platform* NetBeans dapat dengan mudah dikembangkan oleh pihak ketiga secara mudah dan *powerful.*

Pengembangan NetBeans diawali dari Xelfi, sebuah proyek mahasiswa tahun 1997 di bawah bimbingan Fakultas Matematika dan Fisika Universitas Charles, Praha. Sebuah perusahaan kemudian dibentuk untuk proyek tersebut dan menghasilkan versi komersial NetBeans IDE hingga kemudian dibeli oleh Sun Microsystem pada tahun 1999. Sun kemudian menjadikan NetBeans open source pada bulan Juni tahun 2000. Sejak itu komunitas NetBeans terus berkembang.

2.7.1. *Platform* NetBeans

Platform NetBeans adalah *framework* yang dapat digunakan kembali (*reusable*) untuk menyederhanakan pengembangan aplikasi desktop. Ketika aplikasi berbasis *Platform* NetBeans dijalankan, kelas Main dari *Platform*

dieksekusi. Modul-modul yang tersedia ditempatkan di sebuah *registry* di dalam memori, dan tugas *startup* modul dijalankan. Secara umum, kode modul dimuatkan ke dalam memori hanya ketika iadiperlukan.

Aplikasi dapat menginstal modul secara dinamis. Aplikasi dapat memasukkan modul *Update Center* untuk mengijinkan pengguna aplikasi men-download *digitally-signed upgrade* dan fitur-fitur baru secara langsung ke dalam aplikasi yang berjalan. Penginstalan kembali sebuah upgrade atau rilis baru tidak memaksa pengguna untuk men-download keseluruhan aplikasi lagi.

Platform NetBeans menawarkan layanan-layanan yang umum bagi aplikasi desktop, mengijinkan pengembang untuk fokus ke logika yang spesifik terhadap aplikasi. Fitur-fitur yang disediakan oleh *Platform* NetBeans:

1. Manajemen antarmuka (missalnya : menu&toolbar).
2. Manajemen pengaturan pengguna.
3. Manajemen penyimpanan (menyimpan dan membuka berbagai macam data).
4. Manajemen jendela.
5. Wizard framework (mendukung dialog langkah demi langkah).

2.8. Matematika

Menurut **Yudhi Rochman (2008)** Matematika adalah bahasa yang harus dipahami dan dimengerti oleh semua kalangan untuk memudahkan dalam melakukan sebuah proses dan pencapaian hasil. Kata matamatika sudah tidak asing lagi bagi kita, matematika merupakan ratu dari ilmu pengetahuan dimana

materi matematika di perlukan di semua jurusan yang di pelajarai oleh semua orang.

2.8.1. Fungsi dan tujuan matematika

Matematika berfungsi mengembangkan kemampuan menghitung, mengukur, menurunkan dan menggunakan rumus matematika yang diperlukan dalam kehidupan sehari-hari melalui pengukuran dan geometri, aljabar, peluang dan statistik, kalkulus dan trigonometri. Matematika juga berfungsi mengembangkan kemampuan mengkomunikasikan gagasan melalui model matematika yang dapat berupa kalimat matematika dan persamaan matematika, diagram, grafik atau tabel. Sejalan dengan itu tujuan pembelajaran matemátika yang tertuang dalam Kurikulum Tingkat Satuan Pendidikan (KTSP) Tahun 2006 untuk mata pelajaran matematika adalah :

1. Memahami konsep matematika, menjelaskan keterkaitan antar konsep dan mengaplikasikan konsep atau algoritma, secara luwes, akurat , efisien, dan tepat, dalam pemecahan masalah.
2. Menggunakan penalaran pada pola dan sifat, melakukan manipulasi matematika dalam membuat generalisasi, menyusun bukti, atau menjelaskan gagasan dan pernyataan matematika.
3. Memecahkan masalah yang meliputi kemampuan memahami masalah, merancang model matematika, menyelesaikan model dan menafsirkan solusi yang diperoleh.

4. Mengomunikasikan gagasan dengan simbol, tabel, diagram, atau media lain untuk memperjelas keadaan atau masalah.
5. Memiliki sikap menghargai kegunaan matematika dalam kehidupan, yaitu memiliki rasa ingin tahu, perhatian, dan minat dalam mempelajari matematika, serta sikap ulet dan percaya diri dalam pemecahan masalah.

2.8.2. Ruang lingkup

Standar kompetensi matematika merupakan seperangkat kompetensi matematika yang dibukukan dan harus ditunjukkan oleh siswa pada hasil belajarnya dalam mata pelajaran matematika. Standar ini dirinci dalam komponen kompetensi dasar beserta hasil belajarnya, indikator dan materi pokok untuk setiap aspeknya. Pengorganisasian dan pengelompokan materi pada materi didasarkan menurut disiplin ilmunya atau didasarkan menurut kemahiran atau kecakapan yang hendak dicapai. Aspek atau ruang lingkup materi pada standar kompetensi matematika adalah bilangan, pengukuran dan geometri, aljabar, trigonometri, peluang statistik, dan kalkulus.

2.8.3. Materi Pokok dan Sub Pokok

Tabel 2.3 Materi Pembelajaran Matematika SMP

Kelas VII semester 1

Materi Pokok	Sub Materi pokok
Bilangan 1. Memahami sifat-sifat operasi hitung bilangan dan penggunaannya dalam pemecahan masalah.	1.1 Melakukan operasi hitung bilangan bulat dan pecahan. 1.2 Menggunakan sifat-sifat operasi hitung bilangan bulat dan pecahan dalam pemecahan masalah.

Aljabar 2. Memahami bentuk aljabar, persamaan dan pertidaksamaan linear satu variable.	2.1 Mengenali bentuk aljabar dan unsur- unsurnya. 2.2 Melakukan operasi pada bentuk aljabar. 2.3 Menyelesaikan persamaan linear satu variable. 2.4 Menyelesaikan pertidaksamaan linear satu variable.
3. Menggunakan bentuk aljabar, persamaan dan pertidaksamaan linear satu variabel, dan perbandingan dalam pemecahan masalah.	3.1 Membuat model matematika dari masalah yang berkaitan dengan persamaan dan pertidaksamaan linear satu variable. 3.2 Menyelesaikan model matematika dari masalah yang berkaitan dengan persamaan dan pertidaksamaan linear satu variable. 3.3 Menggunakan konsep aljabar dalampemecahan masalah aritmetika sosial yang sederhana. 3.4 Menggunakan perbandingan untuk pemecahan masalah.

Materi 2.4 Pembelajaran Matematika SMP

Kelas VII, Semester 2

Materi pokok	Sub Materi Pokok
Aljabar 4. Menggunakan konsep himpunan dan diagram Venn dalam pemecahan masalah.	1.1 Memahami pengertian dan notasi himpunan, serta penyajiannya. 1.2 Memahami konsep himpunan bagian. 1.3 Melakukan operasi irisan, gabungan, kurang (*difference*), dan komplemen pada himpunan. 1.4 Menyajikan himpunan dengan diagram Venn. 1.5 Menggunakan konsep himpunan dalam pemecahan masalah.
Geometri 5. Memahami hubungan garis dengan garis, garis dengan sudut, sudut dengan sudut, serta menentukan ukurannya.	5.1 Menentukan hubungan antara dua garis, serta besar dan jenis sudut. 5.2 Memahami sifat-sifat sudut yang terbentuk jika dua garis berpotongan atau dua garis sejajar berpotongan dengan garis lain. 5.3 Melukis sudut. 5.4 Membagi sudut.

6. Memahami konsep segi empat dan segitiga serta menentukan ukurannya.	6.1 Mengidentifikasi sifat-sifat segitiga berdasarkan sisi dan sudutnya. 6.2 Mengidentifikasi sifat-sifat persegi panjang, persegi, trapesium, jajargenjang, belah ketupat dan layang-layang. 6.3 Menghitung keliling dan luas bangun segitiga dan segi empat serta menggunakannya dalam pemecahan masalah. 6.4 Melukis segitiga, garis tinggi, garis bagi, garis berat dan garis sumbu.

2.9. UML (Unified Modeling Language)

Menurut **Prabowo Pudjo Widodo dan Herlawati (2001 : 6)** UML adalah singkatan dari Unified Modeling Language yang berarti bahasa pemodelan standar. Menurut **Cholones (2003: Bab 1)** yang dikutip dalam buku tersebut, mengatakan UML sebagai bahasa, berarti UML memiliki sintaks dan semantic. Ketika kita membuat model menggunakan konsep UML dan aturan-aturan yang harus diikuti. Bagaimana elemen-elemen pada model-model yang kita buat berhubungan satu dengan lainnya harus mengikuti standar yang ada. UML bukan hanya sekedar diagram, tetapi juga menceritakan konteksnya.

UML diaplikasikan untuk maksud tertentu, biasanya antara lain untuk :

1. Merancang Perangkat Lunak.
2. Sarana komunikasi antara perangkat lunak dengan proses bisnis.
3. Menjabarkan system secara rinci untuk analisa dan mencari apa yang diperlukan system.
4. Mendokumentasi system yang ada, proses-proses serta pengorganisasiannya.