Sejarah Java

Java dipelopori oleh James Gosling, Patrick Naughton, Chris Warth, Ed Frank, dan Mike Sheridan dari Sun Microsystems, Inc pada tahun 1991. Mereka membutuhkan kurang lebih 18 bulan untuk membuat versi pertamanya. Bahasa ini pada awalnya disebut “Oak” tapi kemudian diubah menjadi “Java” pada tahun 1995 karena nama Oak telah dijadikan hak cipta dan digunakan sebagai bahasa pemrograman lainnya. Antara pembuatan Oak pada musim gugur 1992 hingga diumumkan ke publik pada musim semi 1995, banyak orang yang terlibat dalam desain dan evolusi bahasa ini. Bill Joy, Arthur van Hoff, Jonathan Payne, Frank Yellin, dan Tim Lindholm merupakan kontributor kunci yang mematangkan prototipe aslinya.

Apa itu Teknologi JAVA?

1. **Sebuah Bahasa Pemrograman**

   Sebagai sebuah bahasa pemrograman, Java dapat membuat seluruh bentuk aplikasi, *desktop, web* dan lainnya, sebagaimana dibuat dengan menggunakan bahasa pemrograman konvensional yang lain.

   Java adalah bahasa pemrograman yang berorientasi objek (OOP) dan dapat dijalankan pada berbagai *platform* sistem operasi. Perkembangan Java tidak hanya terfokus oada satu sistem operasi, tetapi dikembangkan untuk berbagai sistem operasi dan bersifat *open source.*

   Platform Java

   Java sebagai platform pengembangan software, secara garis besar dibedakan ke dalam 3 arahan :

   - **J2SE**
     Java 2 Standard Edition mencakup core dari bahasa pemrograman Java, memuat library-library inti yang dibutuhkan seperti IO, Networking dan JDBC.
   - **J2EE**
     Java 2 Enterprise Edition adalah pengembangan Java untuk solusi enterprise, mulai dari aplikasi berbasis Web dengan Servlet dan JSP, aplikasi terdistribusi dengan EJB, sebagaimana aplikasi integrasi enterprise seperti Web Service.
   - **J2ME**
     Java 2 Micro Edition adalah pengembangan Java untuk mobile device, seperti handphone, pocket PC dan PDA. Pengembangan ke arah mobile device ini menuntut Java untuk beradaptasi dengan mesin yang terbatas dalam memory dan processor.

2. **Sebuah *Development Environment***

   Sebagai sebuah peralatan pembangun, teknologi Java menyediakan banyak *tools* : *compiler, interpreter,* penyusun dokumentasi, paket kelas dan sebagainya.

3. **Sebuah Aplikasi**

   Aplikasi dengan teknologi Java secara umum adalah aplikasi serba guna yang dapat dijalankan pada seluruh mesin yang memiliki *Java Runtime Environment* (JRE).

4. **Sebuah *Deployment Environment***

   Terdapat dua komponen utama dari *Deployment Environment*. Yang pertama adalah JRE, yang terdapat pada paket J2SDK, mengandung kelas – kelas untuk semua paket teknologi Java yang meliputi kelas dasar dari Java, komponen GUI dan sebagainya. Komponen yang lain terdapat pada Web Browser. Hampir seluruh Web Browser komersial menyediakan *interpreter* dan *runtime environment* dari teknologi Java.

Mengapa Mempelajari JAVA?

Berdasarkan *white paper* resmi dari SUN, Java memiliki karakteristik berikut :

1. **Sederhana (*Simple*)**
   Bahasa pemrograman Java menggunakan Sintaks mirip dengan C++ namun sintaks pada Java telah banyak diperbaiki terutama menghilangkan penggunaan pointer yang rumit dan *multiple inheritance*. Java juga menggunakan *automatic memory allocation* dan *memory garbage collection*.
2. **Berorientasi objek (*Object Oriented*)**
   Java mengunakan pemrograman berorientasi objek yang membuat program dapat dibuat secara modular dan dapat dipergunakan kembali. Pemrograman berorientasi objek memodelkan dunia nyata kedalam objek dan melakukan interaksi antar objek-objek tersebut.
3. **Terdistribusi (*Distributed*)**
   Java dibuat untuk membuat aplikasi terdistribusi secara mudah dengan adanya *libraries* networking yang terintegrasi pada Java.
4. **Interpreted**
   Program Java dijalankan menggunakan interpreter yaitu *Java Virtual Machine* (JVM). Hal ini menyebabkan *source code* Java yang telah dikompilasi menjadi Java *bytecodes* dapat dijalankan pada platform yang berbeda-beda.
5. **Robust**
   Java mempuyai reliabilitas yang tinggi. Compiler pada Java mempunyai kemampuan mendeteksi error secara lebih teliti dibandingkan bahasa pemrograman lain. Java mempunyai *runtime-Exception handling* untuk membantu mengatasi error pada pemrograman.
6. **Secure**
   Sebagai bahasa pemrograman untuk aplikasi internet dan terdistribusi, Java memiliki beberapa mekanisme keamanan untuk menjaga aplikasi tidak digunakan untuk merusak sistem komputer yang menjalankan aplikasi tersebut.
7. **Architecture Neutral**
   Program Java merupakan *platform independent*. Program cukup mempunyai satu buah versi yang dapat dijalankan pada platform berbeda dengan *Java Virtual Machine*.
8. **Portable**
   Source code maupun program Java dapat dengan mudah dibawa ke platform yang berbeda-beda tanpa harus dikompilasi ulang.
9. **Performance**
   Performance pada Java sering dikatakan kurang tinggi. Namun performance Java dapat ditingkatkan menggunakan kompilasi Java lain seperti buatanInprise, Microsoft ataupun Symantec yang menggunakan *Just In Time Compilers* (JIT).
10. **Multithreaded**
    Java mempunyai kemampuan untuk membuat suatu program yang dapat melakukan beberapa pekerjaan secara sekaligus dan simultan.
11. **Dynamic**
    Java didesain untuk dapat dijalankan pada lingkungan yang dinamis. Perubahan pada suatu *class* dengan menambahkan properties ataupun method dapat dilakukan tanpa menggangu program yang menggunakan *class* tersebut.

Sebagian Fitur dari JAVA

1. Java Virtual Machine (JVM)

JVM adalah sebuah mesin imajiner (maya) yang bekerja dengan menyerupai aplikasi pada sebuah mesin nyata. JVM menyediakan spesifikasi hardware dan platform dimana kompilasi kode Java terjadi. Spesifikasi inilah yang membuat aplikasi berbasis Java menjadi bebas dari *platform* manapun karena proses kompilasi diselesaikan oleh JVM.

Aplikasi program Java diciptakan dengan *file* teks berekstensi *.java*. Program ini dikompilasi menghasilkan satu berkas *bytecode* berekstensi *.class* atau lebih. *Bytecode* adalah serangkaian instruksi serupa instruksi kode mesin. Perbedaannya adalah kode mesin harus dijalankan pada sistem komputer dimana kompilasi ditujukan, sementara *bytecode* berjalan pada *java interpreter* yang tersedia di semua *platform* sistem komputer dan sistem operasi.

2. *Garbage Collection*

Banyak bahasa pemrogaman lain yang mengijinkan seorang pemrogram mengalokasikan memori pada saat dijalankan. Namun, setelah menggunakan alokasi memori tersebut, harus terdapat cara untuk menempatkan kembali blok memori tersebut supaya program lain dapat menggunakannya. Dalam C, C++ dan bahasa lainnya, adalah pemrogram yang mutlak bertanggung jawab akan hal ini. Hal ini dapat menyulitkan bilamana pemrogram tersebut alpa untuk mengembalikan blok memori sehingga menyebabkan situasi yang dikenal dengan nama *memory leaks.*

Program Java melakukan *garbage collection* yang berarti program tidak perlu menghapus sendiri objek – objek yang tidak digunakan lagi. Fasilitas ini mengurangi beban pengelolaan memori oleh pemrogram dan mengurangi atau mengeliminasi sumber kesalahan terbesar yang terdapat pada bahasa yang memungkinkan alokasi dinamis.

3. *Code Security*

Code Security terimplementasi pada Java melalui penggunaan Java Runtime Environment (JRE). Java menggunakan model pengamanan 3 lapis untuk melindungi sistem dari *untrusted Java Code*.

1. Pertama, *class-loader* menangani pemuatan kelas Java ke *runtime interpreter.* Proses ini menyediakan pengamanan dengan memisahkan kelas – kelas yang berasal dari *local disk* dengan kelas – kelas yang diambil dari jaringan. Hal ini membatasi aplikasi Trojan karena kelas – kelas yang berasal dari *local disk* yang dimuat terlebih dahulu.
2. Kedua, *bytecode verifier* membaca *bytecode* sebelum dijalankan dan menjamin *bytecode* memenuhi aturan – aturan dasar bahasa Java.
3. Ketiga, manajemen keamanan menangani keamanan tingkat aplikasi dengan mengendalikan apakah program berhak mengakses sumber daya seperti sistem file, *port* jaringan, proses eksternal dan sistem *windowing*.

Setelah seluruh proses tersebut selesai dijalankan, barulah kode program di eksekusi. Java juga menyediakan beragam teknik pengamanan lain :

1. Bahasa dirancang untuk mempersulit eksekusi kode perusak. Peniadaan *pointer* merupakan langkah besar pengamanan. Java tidak mengenal operasi

pointer. Di tangan pemrogram handal, operasi pointer merupakan hal yang luar biasa untuk optimasi dan pembuatan program yang efisien serta mengagumkan. Namun mode ini dapat menjadi petaka di hadapan pemrogram jahat. Pointer merupakan sarana luar biasa untuk pengaksesan tak diotorisasi. Dengan peniadaan operasi *pointer*, Java dapat menjadi bahasa yang lebih aman.

2. Java memiliki beberapa pengaman terhadap *applet*. Untuk mencegah program bertindak mengganggu media penyimpanan, maka *applet* tidak diperbolehkan melakukan *open, read* ataupun *write* terhadap berkas secara sembarangan. Karena Java *applet* dapat membuka jendela *browser* yang baru, maka jendela mempunyai logo Java dan teks identifikasi terhadap jendela yang dibuka. Hal ini mencegah jendela *pop-up* menipu sebagai permintaan keterangan *username* dan *password*.

Memulai Java

Sebelum memulai untuk belajar java, sebaiknya siapkan dulu tools yang diperlukan yaitu JDK atau Java Development Kit J2SE (Java 2 Standard Edition) dari situs http://java.sun.com.
Mengapa memilih J2SE? karena yang kita pelajari adalah pemrograman java yang dikhususkan untuk komputer desktop. Tidak bisa hanya menggunakan JRE (Java Runtime Environment) Karena JRE tidak menyertakan paket compiler didalamya. Gunakan JDK karena dalam JDK telah lengkap semua yang kita perlukan dalam pembelajaran disini yaitu compiler maupun runtime environmentnya.

Menuliskan source kodenya
Ada beberapa IDE atau development environtment atau tool tambahan yang dapat kita pergunakan untuk menuliskan source code java seperti Eclipse atau Netbeans. Namun karena kita sedang mempelajari dasar pemrogramannya, maka saya berinisiatif menggunakan notepad saja supaya kita lebih mamahaminya.

Program JAVA

```
public class Hello
{
/**
* My first java program
*/
public static void main(String[] args) {

//Menampilkan kata "Hello world" dilayar

System.out.println("Hello world!");
}
}
```