Java API

Yohanes Nugroho

API

- API = Application Programming Interface
- Seperangkat fungsi standar yang disediakan oleh OS atau Bahasa
- Dalam Java, API dimasukkan ke dalam package-package yang sesuai dengan fungsinya

Overview API

- Java mengandung ratusan kelas standar
  - J2SE: Edisi standar
  - J2EE: Edisi enterprise (lebih banyak kelas)
  - J2ME: Subset kelas standar
- Kelas-kelas ini memungkinkan pembuatan program dengan mudah
- API Java cukup lengkap
  - Mulai dari yang sederhana (misalnya struktur data Stack)
  - Sampai yang kompleks (seperti enkripsi dan akses file ZIP)

Memakai API

- Dilakukan dengan mengimpor package/kelas

```
import java.util.Stack;
```

- Ada beberapa kelas bernama sama di package yang berbeda
  - import salah satu dan gunakan nama lengkap untuk yang lain, atau:
  - gunakan nama lengkap semua kelas

API yang akan dibahas

- I/O
- String
- Math
- Utility
- Swing

[1] Input/Output

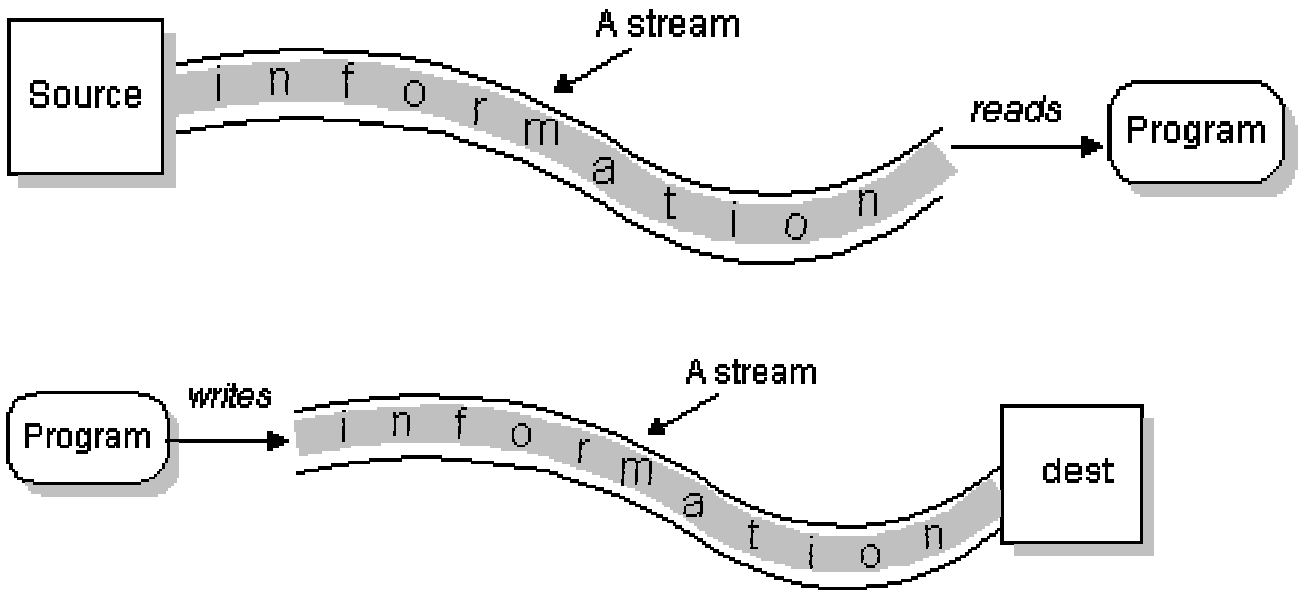
I/O Stream di Java

Overview I/O

- Konsep Stream
- Kelas Abstrak pada java.io
- Reader/Writer
- Input dan Output file
- Wrapper

Konsep Stream

- Stream adalah suatu abstraksi untuk data input dan output
  - Tidak peduli dari mana input berasal atau kemana output akan pergi



Package java.io

- Package java.io berisi kelas yang berhubungan dengan I/O di Java
- Dibagi menjadi beberapa kelas
  - Reader/Writer dan turunannya
  - InputStream/OutputStream dan turunannya
  - I/O Network (socket TCP/IP)
  - Exception

Kelas Abstrak pada java.io

- Ada 4 kelas abstrak dasar
  - `Reader`
  - `Writer`
  - `InputStream`
  - `OutputStream`
- `Reader` dan `Writer` digunakan untuk data Teks
- `InputStream` dan `OutputStream` digunakan untuk data biner

Kelas Reader/Writer

- Kelas Reader/Writer bekerja pada stream teks (bahasa apapun)
  - Disebut juga character stream
  - Menangani konversi teks Unicode secara otomatis
- Jangan dipertukarkan dengan InputStream/OutputStream secara sembarangan
  - Hasilnya bisa error

Diagram Kelas Reader dan Turunannya

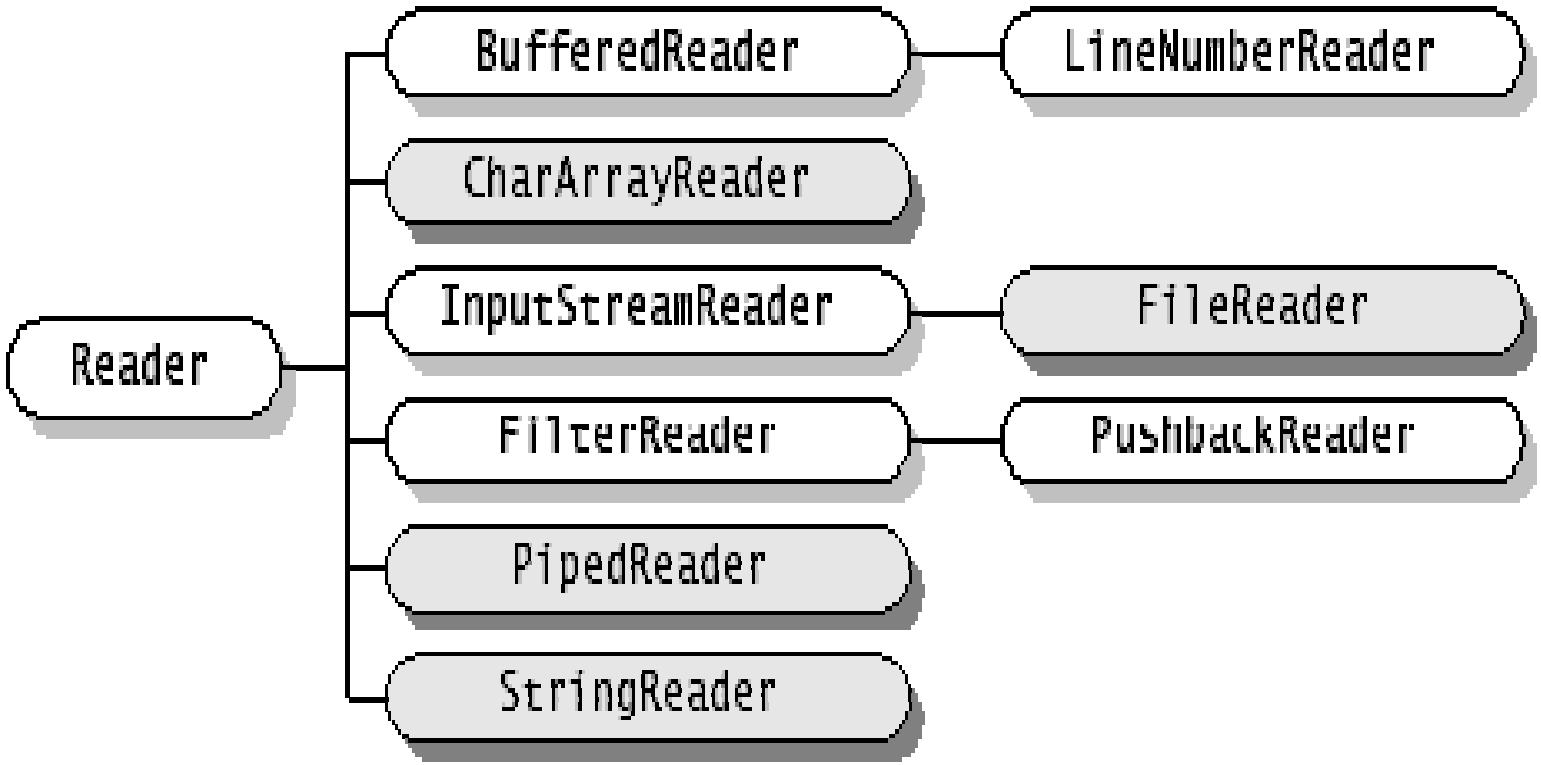
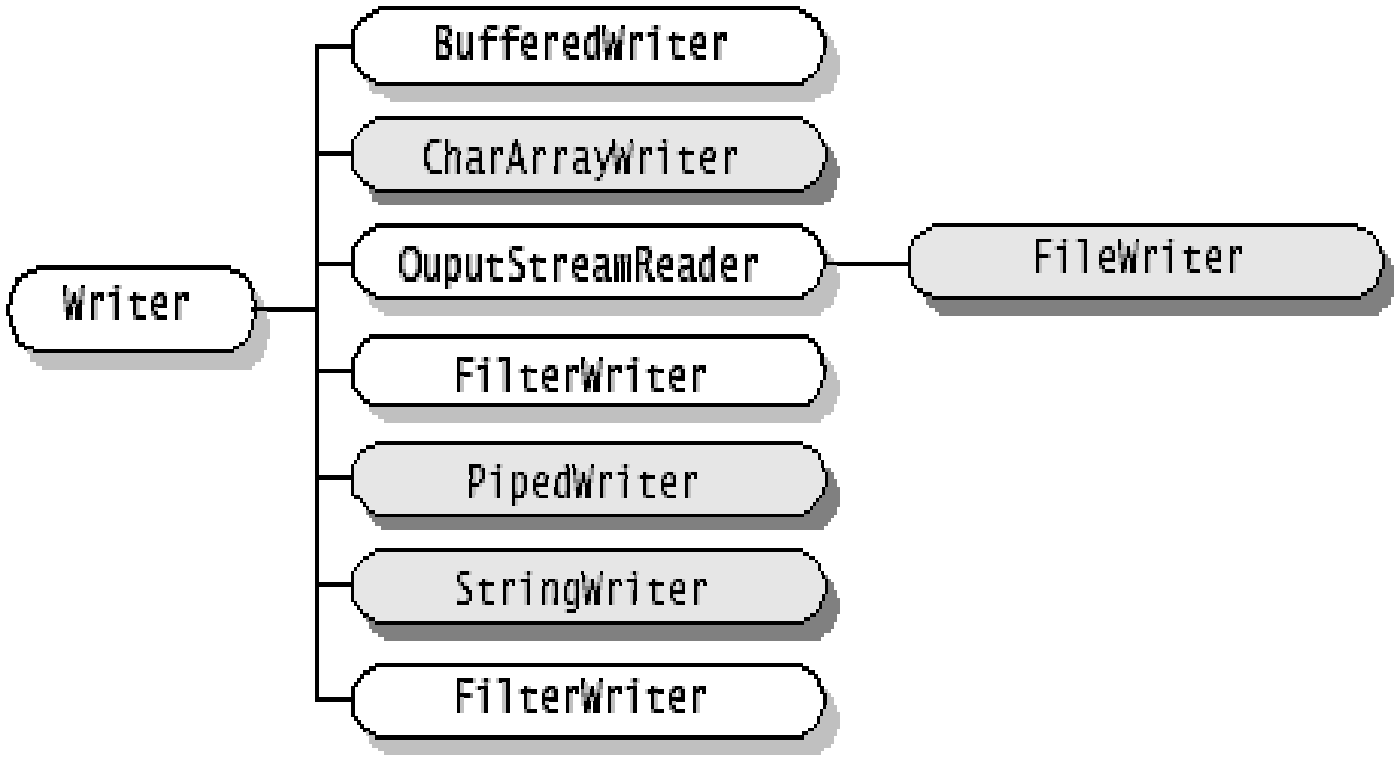


Diagram Kelas Writer dan Turunannya



Kelas InputStream/OutputStream

- `InputStream` dan `OutputStream` digunakan untuk mengakses informasi non teks (biner)
  - disebut juga sebagai byte stream
  - COntoh data biner: File EXE, file GIF
- InputStream/OutputStream Tidak menangani konversi teks Unicode secara otomatis

Diagram Kelas InputStream dan turunannya

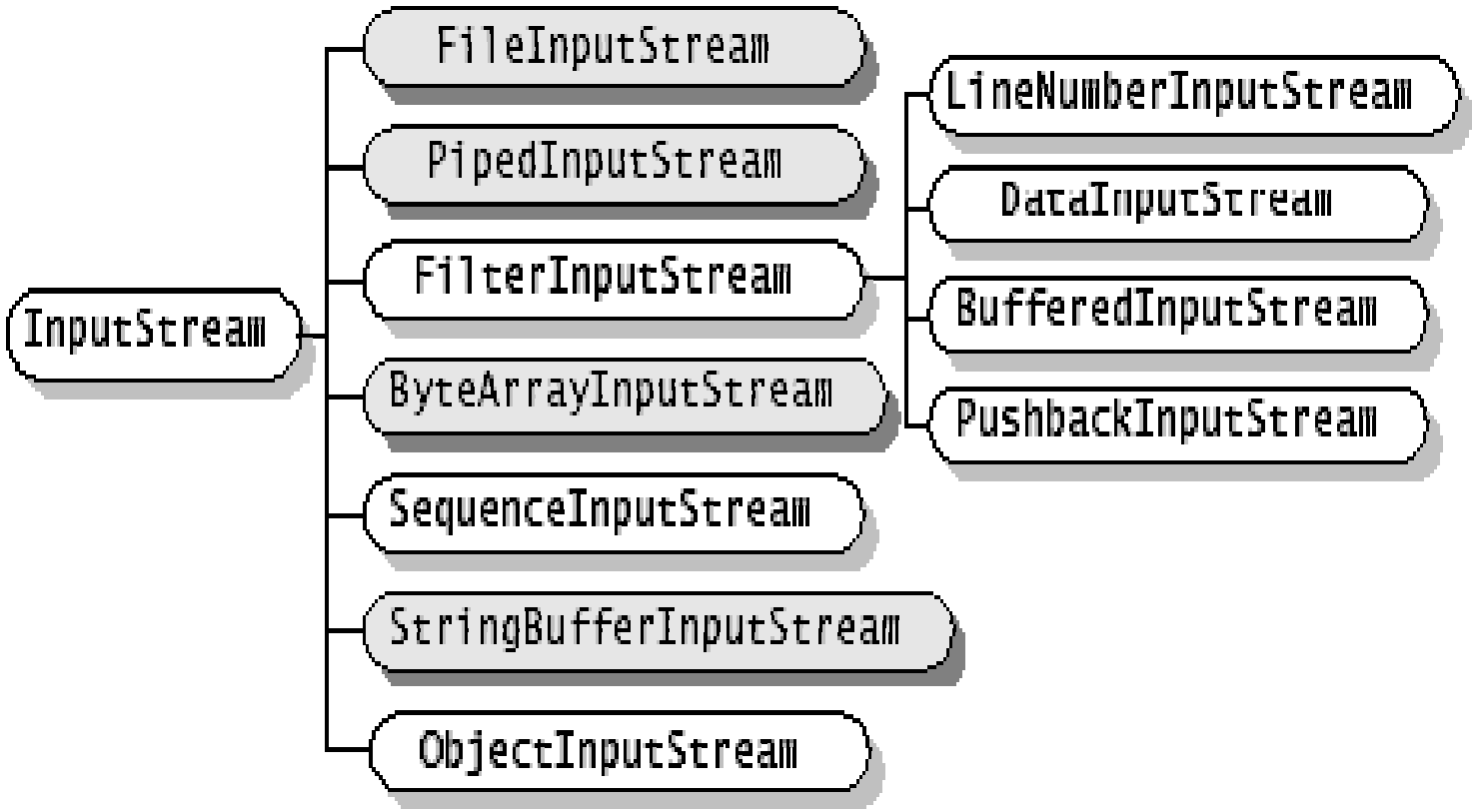
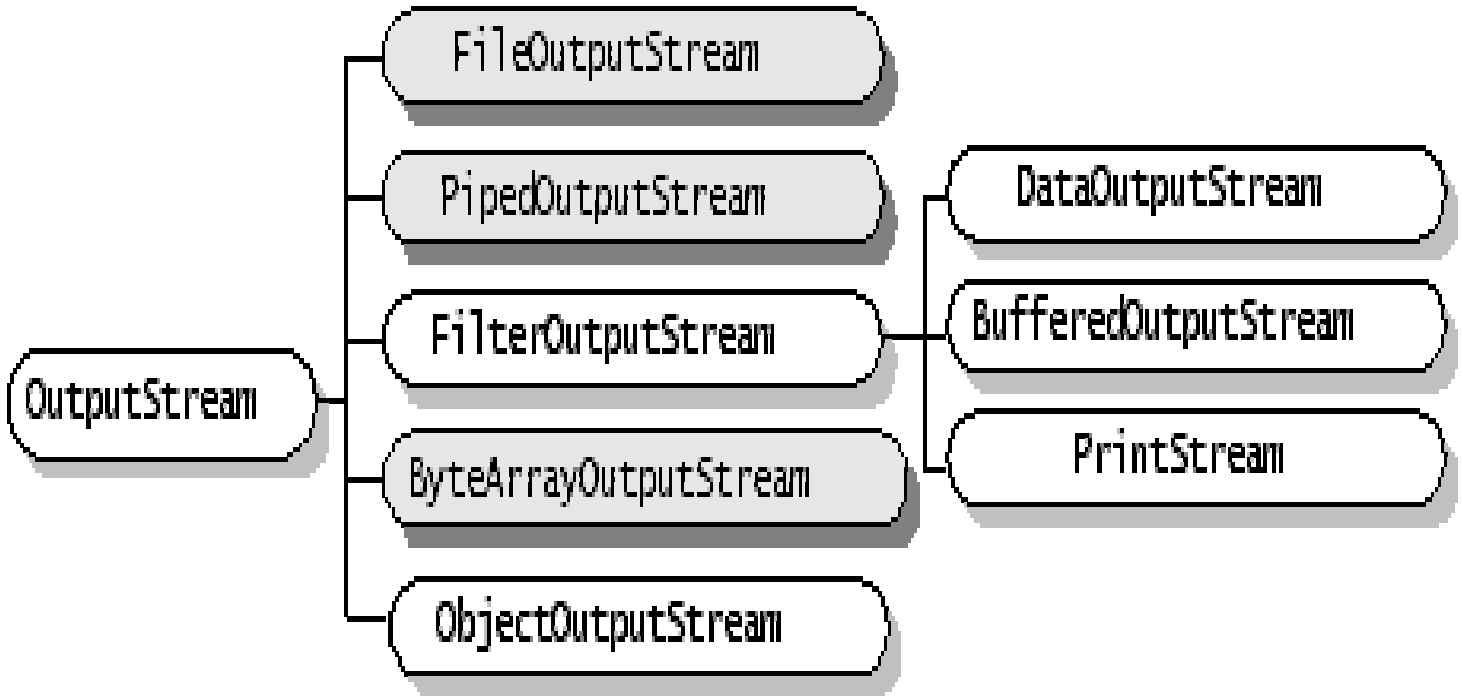


Diagram Kelas OutputStream dan turunannya



Input dan Output File

- Gunakan `FileInputStream` dan `FileOutputStream` untuk membaca/menulis file non teks
- Gunakan `FileReader`/`FileWriter` untuk file teks

Contoh Membuka file untuk dibaca

- cara untuk membuka file (untuk dibaca):

```
FileInputStream s = new
FileInputStream("test.dat");
```

- Cara untuk membaca satu byte:

```
int a = s.read()
```

- Tersedia juga method untuk membaca array of byte
- Prototype read adalah : `int read()`
  - Perlu dicast ke byte untuk membaca sebagai byte
  - Memakai int supaya cukup merepresentasikan -1

Contoh Membuka file untuk ditulis

- Cara untuk membuka file (untuk ditulisi):

```
FileOutputStream out = new
FileOutputStream("test.out");
```

- Intruksi untuk menulis satu byte:
  - `Out.write('a')`
- Tersedia juga method untuk menulis array of byte

Perhatikan:

- Membaca dan menulis selalu perlu `try … catch`
- Method `read()` dan `write()` melempar `Exception java.lang.IOException`
- Kasus khusus: `read()` mengembalikan -1 pada end of file
  - EOF (End Of File atau EndOfStream) tidak menimbulkan exception

Memakai FileReader/FileWriter

- Sama seperti contoh sebelumnya
  - Ganti nama kelas `FileInputStream` dengan `FileReader` dan `FileOutputStream` dengan `FileWriter`
  - Tetap gunakan read/write
- Perhatikan bahwa
  - Nilai kembalian

Wrapper

- Stream dapat dikomposisi atau difilter atau dibungkus untuk menghasilkan stream yang lebih kompleks
- Konsep ini mirip dengan pipe di Unix/Windows
- Contoh:
  - `BufferedReader` yang menyediakan `readLine()`
  - `PipedWriter/PipedReader` untuk mempipe suatu stream
  - `PrintStream/PrintWriter` untuk menulis ke stream dengan mudah

Contoh Wrapper: BufferedReader

- Perhatikan potongan kode berikut:

```
BufferedReader br = new
  BufferedReader(new
  FileReader("hello.txt"));
//membaca 1 baris
String teks = br.readLine();
```

- `BufferedReader` membungkus (wraps) `FileReader` untuk menyediakan method `readLine()`

Contoh Wrapper: PrintWriter

- Perhatikan potongan kode berikut:

```
PrintWriter pr = new
  PrintWriter(new
  FileWriter("hello.txt"));
//Menulis 1 baris
pr.println("Hello ");
```

- `PrintWriter` membungkus (wraps) `FileWriter` untuk menyediakan method `print()`, `println()` untuk semua tipe data dasar

Membaca dari Console [1]

- Java menyediakan `System.in` yang class-nya adalah `InputStream`
  - Membaca teks dari console
- Untuk membaca teks, perlu di-wrap dengan `BufferedReader`
  - Tapi `BufferedReader` hanya bisa me-wrap suatu class turunan `Reader`
- `InputStream` perlu di-wrap dengan `InputStreamReader`

Membaca dari console [2]

- Buat `BufferedReader` yang membungkus `InputStreamReader` yang membungkus `System.in`

```
BufferedReader br = new BufferedReader(new
  InputStreamReader(System.in))
```

- Untuk membaca:

```
String s = br.readLine();
```

- Untuk membaca integer, teks dibaca dengan method yang sama, lalu dikonversi dengan method `Integer.parseInt()`

NIO (Nonblocking I/O)

- JDK 1.4 (nama Kode: Merlin) ke atas menyediakan NIO
  - Ada di package java.nio
- Improvement:
  - Non blocking I/O
  - Buffer
  - Regular Expression

[2] String dan StringBuffer

Penanganan String optimal di Java

String

- Merupakan kelas khusus di Java (ditangani secara transparan)
- Sifatnya immutable (tidak bisa diubah)
- Memiliki berbagai method untuk memiliki manipulasi `String`

Literal `String`

- Harap diingat lagi bahwa:
  - Setiap Literal `String` adalah sebuah objek `String`. Contoh:

```
String teks = "Hello";
System.out.println(teks.length());
```

  - Sama dengan

```
System.out.println("Hello".length());
```

Sifat Immutable `String`

- `String` sebenarnya tidak dapat diubah, namun Java memungkinkan `String` seolah-olah diubah, Contoh:
  - Jika dilakukan ini:

```
String judul = "Judul :";
judul += "Eyes On Me";
```

  - Maka sebenarnya yang dilakukan adalah ini

```
String judul = "Judul :";
judul = judul.concat("Eyes On Me");
```

Operasi `String` Tidak Optimal

- `String` baru diciptakan (string yang lama tetap ada di memori, dan dibuang ketika terjadi garbage collection)
- Untuk operasi yang banyak melibatkan perubahan string, sebaiknya menggunakan `StringBuffer`

StringBuffer

- `StringBuffer` mirip dengan `String`
- Sifatnya mutable
- Tidak ditangani secara transparan oleh Java (harus dilakukan secara manual)
- Lebih cepat untuk manipulasi string yang memerlukan perubahan pada `String`.

Sifat mutable `StringBuffer`

- Untuk mengubah `StringBuffer` tidak perlu objek baru
  - Contoh :

```
StringBuffer nama = new StringBuffer("mataku");
nama.setCharAt(4, 'm');
```

- Untuk mengubah `String` selalu butuh objek baru (objek lama diubah melalui assignment)

Method yang penting

- Beberapa method `String` dan `StringBuffer` yang penting adalah:
  - `length()`: panjang string
  - `replace()`: mengganti suatu karakter
  - `charAt()`: mengakses karakter di posisi tertentu
  - `trim()`: menghilangkan spasi di awal dan di akhir string
- Perhatikan bahwa meskipun namanya sama, sifat keduanya berbeda
  - `String` menciptakan objek baru, sedangkan `StringBuffer` tidak

Membandingkan `String`

- Method `equals()` membandingkan string untuk memeriksa kesamaan
- Method `equalsIngnoreCase()` melakukan hal yang sama, tapi besar kecil huruf tidak diperhatikan
- Method `compareTo()` menghasilkan 0 jika string sama, >0 jika String1>String2 dan <0 jika String1<String2

[3] Kelas Matematik

Hal yang berhubungan dengan matematika

Matematika di Java

- Fungsi matematika ada di package `java.math`
- Meliputi fungsi trigonometri, dan fungsi matematika standar
  - Berisi juga konstanta penting (seperti PI dan e)
  - Sebagian besar hasil method adalah bilangan riil (tipenya double)

Memformat Hasil

- Class math tidak menyediakan cara untuk memformat keluaran (output) sehingga dapat dicetak dengan rapi
- Perlu class `DecimalFormat` untuk melakukan format terhadap output (`DecimalFormat` ada pada package `java.text`)

Beberapa fungsi matematika yang penting

- Method `pow()` untuk pangkat
- Fungsi-fungsi trigonometri (`sin`, `cos`, `tan`)
- Pembulatan hasil, meliputi
  - ke atas : `ceil`, ke bawah: `floor`, dan terdekat : `round`
- Logaritma

Bilangan Acak

- Bilangan acak sangat penting dalam simulasi, permainan, dan enkripsi
- Bilangan acak dihasilkan dengan kelas Random
- Kelas Random tidak berada dalam package `java.math`, tapi dalam kelas `java.util`
  - Contoh:

```
Random r = new Random();
int x = r.nextInt(10); //random 0..9
```

BigInteger dan BigDecimal

- BigInteger
  - Integer presisi tak hingga
  - Berisi semua method untuk operasi integer dan bahkan method untuk mengecek probabilitas suatu bilangan adalah prima
- BigDecimal
  - Bilangan floating point presisi tak hingga
  - Semua method untuk doube/float

[4] Utility

API untuk aneka macam hal

Utility

- Berbagai macam kelas yang tidak cocok dimasukkan ke package tertentu
- Beberapa yang akan dibahas
  - Hashtable
  - Stack
  - Vector
  - Calendar
- Kelas lain yang cukup berguna untuk dipelajari: List, Tree

Hashtable

- Digunakan untuk menyimpan data dengan assosiasi tertentu
  - Misal nama panggilan diassosiasikan dengan nama lengkap
- Memetakan suatu nilai String dengan suatu Objek tertentu (Objek apa saja, termasuk juga Objek String)
- Method yang dipakai adalah `put` dan `get`

Contoh

```
import java.util.Hashtable;
class HashtableDemo {
   public static void main(String argv[]) {
      Hashtable h  = new Hashtable();
      h.put("Linux", "Torvalds");
      h.put("Windows", "Microsoft");
      String nama = (String) h.get("Linux");
      if (nama!=null) {
         System.out.println(nama);
      }
   }
}
```

Stack

- Struktur data LIFO (Last In First Out)
  - Data yang masuk pertama akan keluar terakhir
- Memakai Method **push**, untuk meletakkan satu item di Stack dan **pop** untuk mengeluarkan satu item dari stack
  - Ada juga method `peek()` untuk mengintip top of stack dan `search` untuk mencari elemen di Stack()

Contoh Stack

```
import java.util.Stack;
class StackDemo {
public static void main(String argv[]) {
        Stack s = new Stack();
        s.push("Salamku Kawan");
        s.push("Jangan Takut Gelap");
        s.push("Gembira berkumpul");
        while (!s.empty()) {
            System.out.println(s.pop());
        }
}
}
```

Vector

- Struktur data seperti array
- Sifatnya dinamis (ukurannya tidak tetap), berubah sesuai dengan elemen yang ditambahkan ke (atau dihapus dari) Vector tersebut
- Dapat dimanipulasi dengan mudah (elemen-elemennya dapat di add, remove, atau diubah dengan mudah)

Contoh Vector

```
import java.util.Vector;
import java.util.Enumeration;
class VectorDemo {
public static void main(String argv[]) {
        Vector v = new Vector();
        /* tambahkan elemen */
        v.add("Pelangiku");
        v.add("Andai Aku Besar Nanti");
        v.add("Dua Balerina");
        for (int i=0; i<v.size(); i++){
            System.out.println(" - " + v.elementAt(i));
        }
}
}
```

Calendar

- Dipakai untuk menangani perhitungan kalender Masehi (Gregorian)
  - Dirancang untuk mendukung kalender lain
- Menyediakan informasi mengenai tanggal, (date) dan waktu (time) saat ini (misalnya hari apa, bulan ke berapa, hari keberapa dalam tahun ini)
- Menyediakan sarana untuk penghitungan waktu

Contoh Calendar

```
import java.util.Calendar;
class Kalender {
public static void main(String argv[]) {
  Calendar c = Calendar.getInstance();
  System.out.println("Hari ini :");
  System.out.println("Hari ke“
+c.get(Calendar.DAY_OF_WEEK)+" dalam
minggu ini");
}
}
```

[5] JFC dan Swing

Membuat GUI di Java

Overview JFC/Swing

- Sekilas mengenai JFC/Swing
- Konsep-konsep
  - Container
  - Komponen
  - Layout management
  - Listener
- Contoh

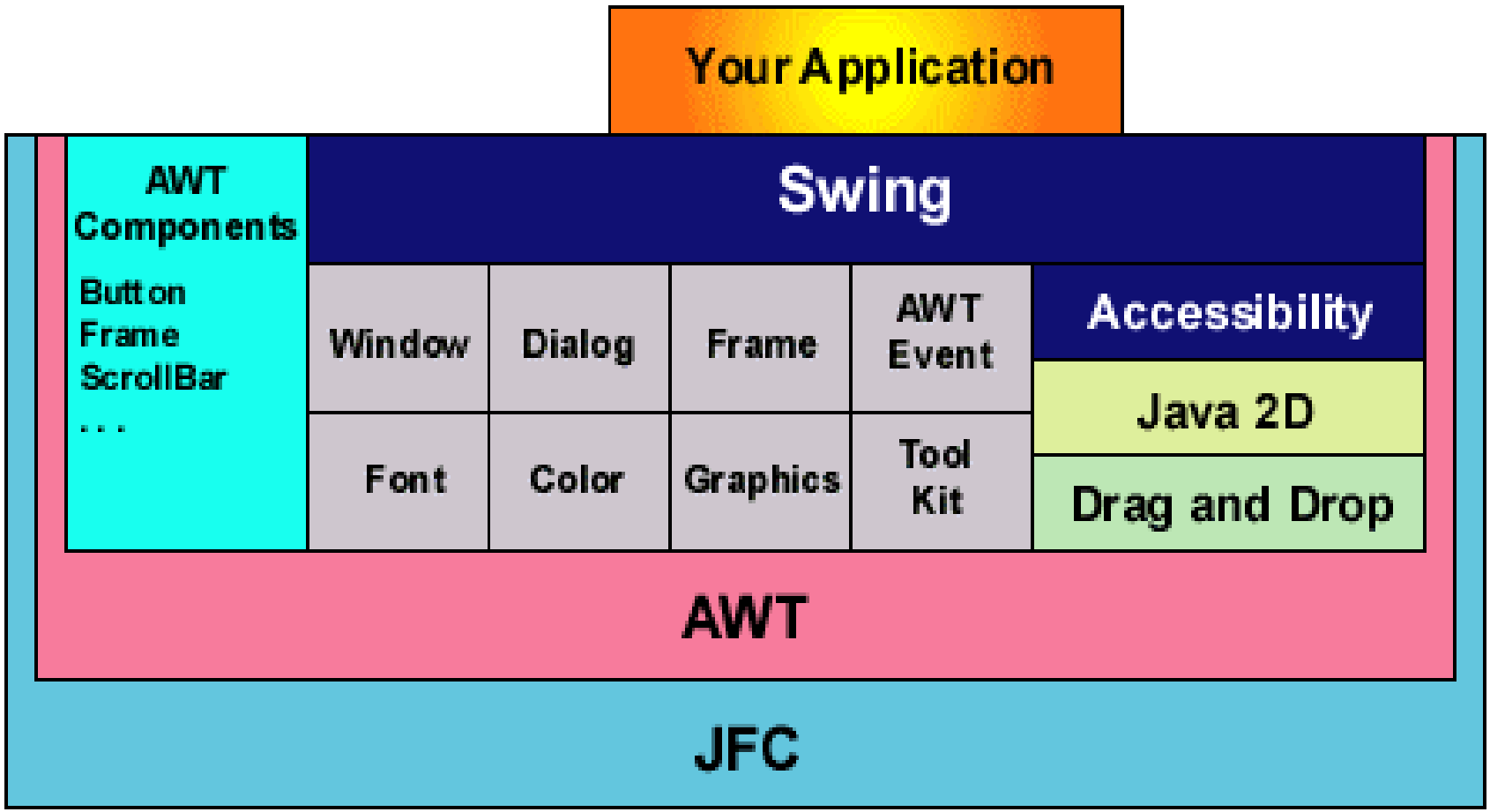
JFC dan Swing

- JFC – Java™ Foundation Classes
- Terdiri dari fitur-fitur untuk membangun GUI
- Diimplementasikan sepenuhnya dalam Java
- Swing adalah nama kode (codename) untuk proyek yang mengembangkan komponen JFC pertama
- Nama Swing biasa digunakan untuk menyebut komponen baru dan API yang berkaitan dengan komponen tersebut

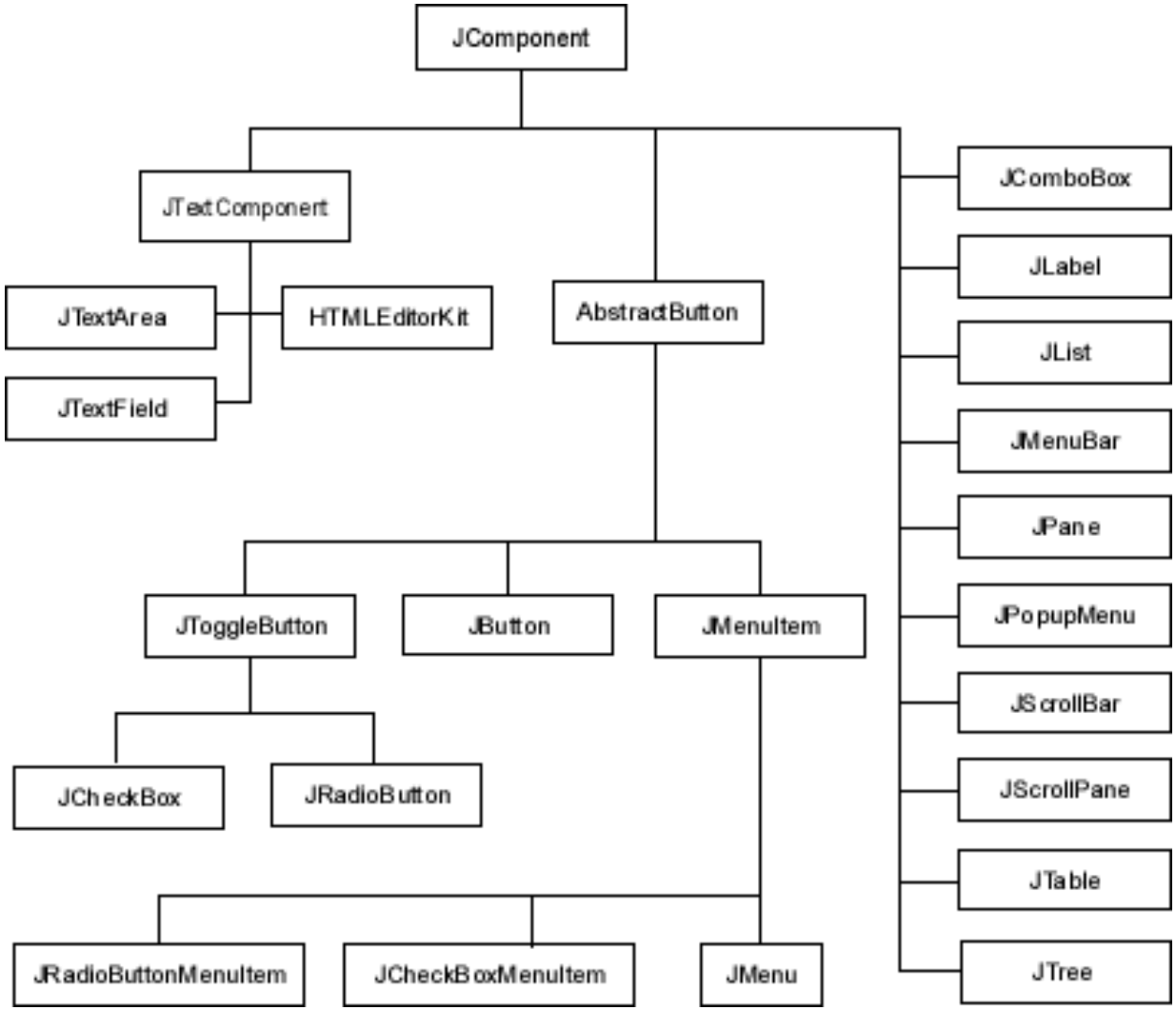
Fitur Swing

- Komponen: Dialog, Tabbed Pane, Button, File Chooser,…
- Pluggable Look and Feel
- Accessibility API: Screen Readers, Braile Display
- Java 2D API
- Drag and Drop API

Arsitektur Aplikasi GUI



Komponen UI


JComponent
JTextComponent
JTextArea
HTMLEditorKit
JTextField
AbstractButton
JToggleButton
JButton
JMenuItem
JCheckBox
JRadioButton
JRadioButtonMenuItem
JCheckBoxMenuItem
JMenu
JComboBox
JLabel
JList
JMenuBar
JPane
JPopupMenu
JScrollBar
JScrollPane
JTable
JTree

Komponen Swing

- Swing menyediakan banyak komponen standar
- Komponen ini dibangun dengan konsep MVC
- Swing menyediakan Container yang bisa menerima komponen di dalamnya
  - Top Level, Intermediate Level, Komponen

Container

- Turunan dari `java.awt.Container`
- Container merupakan komponen yang dapat berisi komponen lain
  - Contoh: Label di dalam Window
- Menggunakan Layout Manager to memposisikan dan mengatur ukuran komponen di dalamnya
  - Bisa null yang artinya posisi dan ukuran komponen diberikan oleh programmer

Top Level Container

- Ada 3 top level container dalam Swing:
  - JFrame
  - JDialog
  - JApplet
- Untuk dapat tampil di layar, setiap komponen GUI harus menjadi bagian dari containment hierarchy, dengan Top Level Container sebagai root
- Setiap top level container memiliki content pane yang mengandung komponen tampak (visible) dalam GUI

Contoh

```
import javax.swing.*;

public class HelloWorldSwing {

  public static void main(String[] args) {
    JFrame frame = new JFrame("HelloWorldSwing");
    JLabel label = new JLabel("Hello World");
    frame.getContentPane().add(label);

   frame.setDefaultCloseOperation(JFrame.EXIT_ON_CL
   OSE);
    frame.pack();
    frame.setVisible(true);
  }
}
```

JDialog

- Bergantung pada frame
  - Persis seperti dialog di Windows
- Dialog bisa bersifat modal
- Ada beberapa dialog standar: JProgressBar, JFileChooser, JColorChooser, ...
- JOptionPane bisa digunakan untuk menciptakan dialog modal sederhana
  - Meminta input, memberikan info, dll

JComponent

- Merupakan kelas dasar dari semua komponen di Swing, kecuali Top Level Container
  - JLabel, JButton, …
- Harus diletakkan di sebuah container
- Setiap JComponen juga adalah container (misalnya Button bisa mengandung Teks dan Ikon)

Fitur JComponent

- Pluggable Look and Feel
- Keystroke handling
- Tooltip support
- Accessibility
- Infrastruktur untuk painting
- Mendukung border

Intermediate Level Container

- Dikenal juga sebagai pane atau panes
- Memudahkan peletakan komponen lain
- Juga bisa mengatur penampilan komponen:
  - JScrollPane
  - JTabbedPane
- Default layout manager adalah Flow Layout

Layout Manager

- Manajemen Layout: Proses menentukan ukuran dan posisi komponen
- Dapat dilakukan dengan absolute positioning
  - Ukuran setiap komponen harus ditentukan
  - Tidak bisa menyesuaikan diri ketika diresize
  - Tidak bisa menyesuaikan dengan perbedaan sistem (misalnya ukuran font yang berbeda)

Contoh Layout Manager

- Grid
  - Komponen dalam tabel
- Flow Layout
  - Komponen "mengalir" ke kanan lalu ke baris berikut
  - Komponen yang ditambahkan akan berada di kanan, lalu (jika tidak muat) di baris berikut

Konsep MVC

- Setiap komponen Swing memakai konsep MVC (Model View Controller)
  - View dan Controller digabung
- Model memungkinkan kita membuat data dari berbagai sumber data (misal data tabel dapat diambil dari database atau dari file teks)

Event Handling

- Memakai Konsep Listener
- Model mengimplementasikan method untuk menambah dan menghapus listener
- Lightweight Notification
  - Hanya memberi tahu bahwa ada event
  - Listener yang akan mencari tahu perubahannya (contoh: scrollbar di drag)
- Stateful Notification
  - Event memberitahukan perubahannya
  - Untuk data model yang kompleks
  - Contoh: perubahan di kolom tabel

Listener

- Event dipropagasikan dengan cara memanggil method pada listener
- Listener adalah kelas yang mengimplementasikan interface listener tertentu
- Kita dapat membuat Anonymous Class untuk listener

Penutup Mengenai Swing

- Swing adalah suatu arsitektur yang dibangun di atas konsep OO yang baik
- Swing dibangun agar portable dan usable
- Untuk saat ini, swing memang lambat karena ditulis 100% dalam Java (ditambah dengan overhead OO)