ANALISIS PERBEDAAN PENGGUNAAN KELAS BUFFEREDREADER DAN KELAS SCANNER DALAM PROSES INPUT KEYBOARD PADA PEMROGRAMAN JAVA BERBASIS TEXT

Dikwan Moeis [1)], Sry Yunarti [2)]
[1)]Sistem Informasi, STMIK Profesional Makassar
email: dikwan_moeis@stmikprofesional.ac.id
[2)]Sistem Informasi, STMIK Profesional Makassar
email: sry_yunarti@stmikprofesional.ac.id

Abstrak

This study aims to provide a clear picture of the location of the differences in functions and the use of source code from the BufferedReader class and the Scanner class in the input process through the keyboard in Java programming. To know the location of the differences in functions and the use of source code of the two classes, the method used in this study is to make a test design as a reference in the analysis, then the design is poured in the form of a simple program prototype. This study obtained the results in the form of tables, namely the table of differences in the BufferedReader class and the Scanner class, and a prototype of a simple program for the input process through the keyboard.

Keywords: Text Based Java, BufferedReader, Scanner, Input Keyboard

A. PENDAHULUAN

Dalam dunia pemrograman komputer, dikenal banyak bahasa pemrograman, beberapa diantaranya yang sudah terkenal adalah bahasa C, Visual Basic .Net, Java, PHP dan lain-lain. Di antara banyaknya bahasa pemrograman yang telah dikenal tersebut, Java merupakan salah satu bahasa pemrograman yang paling banyak diminati dan umum digunakan untuk membuat berbagai aplikasi mulai dari aplikasi *desktop, mobile* hingga berbasis *web*. Java dibuat pertama kali oleh sekelompok insinyur yang dipimpin oleh Patrick Naughton dan James Gosling dari Sun Microsystems pada tahun 1991. Java dikembangkan untuk memenuhi kebutuhan akan sebuah bahasa komputer yang ditulis satu kali dan dapat dijalankan dibanyak sistem komputer yang berbeda tanpa melakukan perubahan kode atau istilahnya *multiplatform*. Versi pertama yang tersedia untuk publik Java (Java 1.0) dirilis pada tahun 1995. Pada tahun 2006 Sun mulai membuat Java tersedia di bawah lisensi GPL (*General Public License*) dan pada tahun 2010 Sun Microsystems diakuisisi oleh Oracle Corporation.

Kelebihan utama Java adalah metode pemrogramannya yang berorientasi objek murni, dibuat berdasarkan kemampuan-kemampuan terbaik bahasa pemrograman objek sebelumnya (C++,

Ada, Simula). Java memiliki *interpreter* disebut *Java Virtual Machine* (JVM) yang memungkinkan sebuah aplikasi bisa dijalankan di atas *platform* apa saja sepanjang pada mesin tersebut dipasang JVM. Fitur unggulan lainnya adalah kemampuan Java dalam mengurangi beban pengelolaan memori untuk menghindari situasi memoy *leaks*, fitur ini dikenal dengan istilah *Garbage Collection*.

Dari beberapa kelebihan Java tersebut, banyak pemula dari kalangan umum, mahasiswa dan akademisi menjadikan dasar pertimbangan untuk memilih bahasa pemrograman ini. Namun, bagi pemula yang hendak mempelajari Java seringkali mengalami kesulitan dalam memahami alur dan konsep pemrogramannya. Berbeda dengan pemrograman prosedural yang relatif mudah dipahami, pada pemrograman Java seorang pemula perlu memahami terlebih dahulu mengenai alur dan konsep pemrograman berorientasi objek sebagai dasar pengetahuan untuk bekerja dalam lingkungan pemrograman Java.

Mempelajari pemrograman Java memiliki tantangan tersendiri, dibutuhkan ketekunan dan kesabaran. Sebagai langkah awal untuk memudahkan pemahaman dalam belajar, seorang pemula dituntut untuk memahami konsep pemrograman berorientasi objek, struktur penulisan kode dan sintaks. Selanjutnya, mampu menulis kode-kode program dasar seperti menampilkan teks, penggunaan variabel, melakukan operasi aritmatika dasar dan melakukan *input* dari *keyboard*.

Pada pemrograman Java berbasis teks, menulis kode-kode program dasar untuk melakukan *input* dari *keyboard* dapat dilakukan dengan menggunakan kelas BufferedReader atau kelas Scanner. Kelas BufferedReader berasal dari paket *Java Input Output* (java.io) dan kelas Scanner berasal dari paket *Java Utility* (java.util). Dalam menerapkannya pada program sederhana, seorang pemula biasanya kesulitan membedakan kedua kelas tersebut dan pada akhirnya dibuat bingung dalam menentukan kelas yang akan digunakannya nanti.

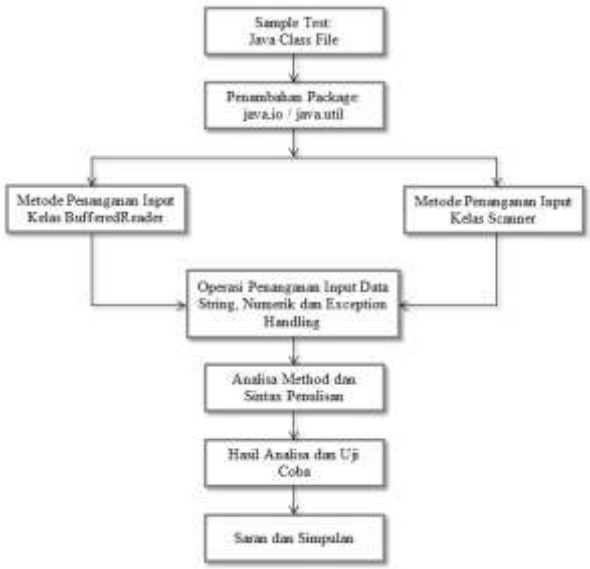
Dari uraian yang telah dipaparkan, penulis bermaksud melakukan penelitan untuk mengetahui secara pasti letak perbedaan fungsi dan kode sumber (*source code*) dari kedua kelas tersebut, khususnya dalam melakukan *input* dari *keyboard*.

B. METODE PENELITIAN

1. Rancangan dan Tahapan Uji Coba

Untuk mengetahui perbedaan kelas *BufferedReader* dan kelas *Scanner* dalam menangani *input*, diperlukan rancangan uji

coba sebagai acuan dalam analisa. Alur rancangan uji coba yang digunakan tampak pada gambar 1 sebagai berikut:



Gambar 1 Alur Rancangan Uji Coba

Dalam tahapan uji coba ini, kita membuat beberapa program sederhana untuk tiap-tiap kelas yang akan digunakan untuk membandingkan kemampuan dari kelas-kelas tersebut dalam menangani *input*. Adapun tahapannya sebagai berikut:

1. Membuat *class* program pada Java.
2. Menambahkan *package* yang dibutuhkan oleh kelas dan mendeklarasikan objek yang akan digunakan untuk menangani *input*.
3. Penggunaan metode yang digunakan dalam menangani *input*.
4. Melakukan operasi *input* untuk data *string*, data numerik dan penanganan *exception* (*exception handling*).
5. Menganalisa sintaks, *method*, cara menangani *input* dan kemampuan menangani *exception* dari masing-masing kelas, kemudian membandingkannya, selanjutnya dicatat bila ditemukan perbedaan-perbedaan yang berkenan dengan hal tersebut.

2. Instrumen Penelitian

Instrumen penelitian berupa perangkat lunak seperti sistem operasi, *editor* dan bahasa pemrograman. Secara rinci *editor* yang digunakan adalah Netbeans IDE, sistem operasi menggunakan Microsoft Windows 7 Ultimate dan bahasa pemrograman Java Standard Edition.

Berdasarkan variabel penelitian yang berupa kelas BufferedReader dan kelas Scanner, maka instrumen penelitian yang menjadi fokus penelitian yaitu *Syntax* dan *Method*.

3. Tahapan Penelitian

Tahapan dalam penelitian ini adalah sebagai berikut:

1. Pengumpulan Data.
   Penulis melakukan studi pustaka untuk mempelajari tentang kelas-kelas *input* yang ada pada pemrograman Java Standard Edition.
2. Analisis Referensi

Penulis melakukan analisis referensi sesuai data dan permasalahan yang telah dikumpulkan sebelumnya.

3. Membuat Program Uji Coba

   Membuat beberapa program uji coba untuk keperluan analisis perbandingan guna mendapatkan perbedaan untuk tiap tiap kelas *input*.

4. Pengujian

   Dilakukan pengujian untuk memperlihatkan hasil dari perbandingan.

5. Penyusunan Laporan

   Setelah semuanya selesai, maka selanjutnya adalah penyusunan laporan hasil penelitian.

C. HASIL PENELITIAN

Setelah melakukan uji coba di laboratorium komputer, penelitian ini mendapatkan hasil yang berupa tabel, yaitu tabel perbedaan penggunaan kelas BufferedReader dan kelas Scanner. Uji coba dilakukan dengan membuat contoh-contoh program sederhana dari masing-masing kelas yang kemudian dilakukan perbandingan untuk melihat kemampuan masing-masing kelas dalam menangani *input*. Untuk melihat perbedaan dari kedua kelas tersebut, perhatikan Tabel 1 di bawah ini:

Tabel 1 Perbedaan Kelas BufferedReader dan Kelas Scanner

No.	Jenis Pembanding	Keterangan	
		BufferedReader	**Scanner**
1.	Package (Paket)	- java.io.BufferedReader - java.io.InputStreamReader - java.io.IOException	Java.util.Scanner
2.	Stream	System.in	System.in
3.	Bentuk deklarasi objek	BufferedReader varObj = new BufferedReader(new InputStreamReader(System.in))	Scanner varObj = new Scanner(System.in)
4.	Penanganan masukan data string	Menggunakan metode readLine()	Menggunakan metode nextLine()
5.	Penanganan masukan data numerik	Menggunakan metode readLine(), namun membutuhkan konversi data ke tipe data primitif yang digunakan.	Tidak membutuhkan konversi. Dalam kelas Scanner, sudah tersedia metode-metode yang berfungsi untuk menangani *input* berupa data numerik, seperti nextByte, nextInt(), nextDouble

			dan sebagainya.
6.	Exception Handling	- Mampu melakukan throws IOException - Bisa menggunakan blok try – catch dalam menangani exception	- Tidak melakukan throws IOException. - Bisa menggunakan blok try – catch dalam menangani exception.

D. PEMBAHASAN

Kelas BufferedReader dan kelas Scanner merupakan dua kelas yang bisa digunakan untuk membaca *file* ataupun *input* dari *user* melalui *command prompt* dalam pemrograman Java berbasis teks. Namun, diantara keduanya ternyata terdapat perbedaan yang signifikan dan oleh karena itu perlu untuk diketahui. Salah satu perbedaan utama adalah bahwa kelas BufferedReader hanya dapat membaca *String*, namun kelas Scanner dapat membaca *String* dan tipe data yang lainnya seperti *long*, *double*, *float*, *short* dan *int*. Perbedaan dalam hal fungsional ini memberikan beberapa perbedaan lain dalam penggunaannya.

Pada kesempatan kali ini, tidak hanya akan dibahas mengenai perbedaan antara keduanya saja. Tapi, akan dijelaskan juga mengenai contoh penggunaan keduanya dalam program.

1. Kelas BufferedReader

BufferedReader adalah kelas yang memungkinkan pembacaan data dari peranti yang berbasis karakter, misalnya dari berkas teks atau dari *keyboard*. Salah satu metode penting dalam kelas BufferedReader yaitu *readLine()*. Metode ini memungkinkan pembacaan sebuah baris teks. Dalam java, *input console* dilakukan melalui pembacaan terhadap *stream System.in*, maka untuk mendapatkan karakter-karakter yang dimasukkan melalui keyboard ke dalam layar *console*, kita perlu membungkus *System.in* di dalam objek BufferedReader. Hal ini dilakukan untuk membentuk *stream* karakter karena *System.in* sebenarnya merupakan *stream byte*. Bentuk *constructor* dari BufferedReader adalah sebagai berikut:

BufferedReader(Reader inputReader)

inputReader adalah *stream* yang akan dihubungkan dengan *instance* atau objek dari kelas BufferedReader yang dibuat. Karena *Reader* merupakan kelas abstrak, maka kita perlu mencari kelas turunannya yang berupa kelas konkrit. Salah satunya adalah kelas *InputStreamReader*, yang

dapat mengkonversi *byte* ke karakter. Sekarang, agar objek dari *InputStreamReader* dapat dihubungkan dengan *System.in*, kita perlu menggunakan bentuk *constructor* sebagai berikut:

InputStreamReader(InputStream inputStream)

dalam hal ini, *inputStream* dapat kita isi dengan *System.in*. Dengan demikian, untuk membuat objek BufferedReader yang dapat terhubung dengan *keyboard*, kita perlu menggunakan kode berikut:

```
BufferedReader dataIn = new BufferedReader(
    new InputStreamReader(System.in));
```

gaya penulisan kode di atas dapat kita ganti seperti berikut ini:

```
InputStreamReader sr = new InputStreamReader(System.in);
BufferedReader dataIn = new BufferedReader(sr);
```

pada tahap ini, objek dataIn sudah siap digunakan untuk melakukan proses *input*, yaitu dengan melakukan pemanggilan terhadap metode *read()* maupun *readLine()*.

2. Kelas Scanner

Scanner adalah kelas dalam paket *java.util* yang digunakan untuk mendapatkan *input* dari tipe data primitif seperti *int*, *double*, *string* dan lain-lain. Ini adalah cara termudah untuk membaca *input* dalam pemrograman Java, meskipun tidak terlalu efisien, namun jika kita menginginkan metode *input* untuk skenario di mana waktu merupakan kendala seperti dalam pemrograman kompetitif.

Secara *default*, kelas scanner bekerja dengan cara memecah *input* menjadi token yang dibatasi oleh karakter spasi, hal ini menyediakan banyak metode untuk membaca dan melakukan parsing pada berbagai nilai primitif. Adapun langkah-langkah menggunakan kelas scanner adalah sebagai berikut:

1. Melakukan *import* kelas scanner yang terdapat pada paket *java.util*. Bentuk penulisannya adalah:

   ```
   import java.util.Scanner;
   ```

2. Membuat objek referensi sebagai media penginputan data. Bentuk penulisannya adalah:

   ```
   Scanner dataIn = new Scanner(System.in);
   ```

3. Memanggil method khusus untuk melakukan inputan data melalui objek referensi yang tadi dibuat.

Tabel 2 berikut ini merupakan daftar beberapa *method input* yang biasa digunakan pada kelas scanner.

Tabel 2 Method Input Pada Kelas Scanner

Method	Berfungsi Untuk Menampung
nextBigDecimal()	Nilai bil. Pecahan BigDecimal
nextBigInteger()	Nilai bil. bulat BigInteger()
nextBoolean()	Nilai berupa Boolean
nextByte()	Nilai bilangan bulat byte
nextDouble()	Nilai bilangan pecahan double
nextFloat()	Nilai bilangan pecahan float
nextInt()	Nilai bilangan bulat int
nextLine()	Data berupa String
nextLong()	Nilai bilangan bulat long
nextShort()	Nilai bilangan bulat short

E. KESIMPULAN DAN SARAN

Kesimpulan

Berdasarkan hasil penelitian yang telah dilakukan, maka dapat ditarik kesimpulan sebagai berikut:

1. Dari penelitian ini dihasilkan beberapa bentuk program sederhana yang digunakan untuk melihat perbedaan penggunaan kelas BufferedReader dan kelas Scanner dalam menangani *input*.
2. Perbedaan pada penggunaan kelas BufferedReader dan kelas Scanner terletak pada jenis *package* yang disertakan, cara mendeklarasikan objek dan *method*, khususnya *method* yang digunakan dalam menangani masukan data berupa data *string* dan data numerik.

Saran

Untuk perbaikan dan penelitian selanjutnya, beberapa saran yang dapat diberikan:

1. Penulis menyarankan kepada pembaca khususnya para pelajar pemrograman Java pemula untuk dapat membedakan dan menggunakan dengan tepat kelas yang akan dipakai dalam menangani *input*.
2. Penelitian ini masih berkelanjutan, sehingga bagi para peneliti dibuka lebar kemungkinan untuk melanjutkannya. Penulis menerima saran yang membangun agar hasil penelitian ini lebih baik.

F. DAFTAR PUSTAKA

[1] Kadir, Abdul. 2004. *Dasar Pemrograman Java 2*. Yogyakarta. Penerbit Andi.

[2] Kadir, Abdul. 2012. *Algoritma dan Pemrograman Menggunakan Java*. Yogyakarta. Andi Publisher.

[3] Raharjo, Budi., Heryanto, Imam dan Haryono, Arif. 2007. *Mudah Belajar Java*. Bandung. Informatika Bandung.

[4] Suarga. 2009. *Dasar Pemrograman Komputer Dalam Bahasa Java*. Yogyakarta. Andi Publisher.

[5] Darmanto, Eko dan Utomo, Andy Prasetyo. 2011. *Analisa Perbedaan Penggunaan Kontrol ADO dan DAO dalam Pemrograman Database Berbasis Server*. Jurnal Sains dan Teknologi, 4 (1). ISSN 1979-6870.

[6] Selamet, Rachmat. 2017. *Pengaksesan File di Java*. Jurnal Media Informatika, Volume 16 No. 1 Tahun 2017.

[7] Syamsudin, Ahmad. 2020. *Analisis Kesalahan Coding Pemrograman Java Pada Matakuliah Algoritma Pemrograman Mahasiswa Tadris Matematika IAIN Kediri*. Jurnal Factor M, Volume 2 No. 2, Juni 2020.

[8] Hardiyana, Bella dan Nopandi, Yayang. 2016. *Rancang Bangun Aplikasi Pembelajaran Pemrograman Berorientasi Objek Dengan Bahasa Java*. Jurnal Teknologi dan Informasi Volume 6 No. 1 Tahun 2016.

[9] Java Hungry, 2019. *Java Developers Tutorials and Coding*. [Online] Available at: https://javahungry.blogspot.com/2018/12/difference-between-bufferedreader-and-scanner-in-java-examples.html, [Diakses 12 Januari 2021].

[10] CODE2SUCCEED, 2019. *Tutorial and Coding. Help You Grow*. [Online] Available at: http://www.code2succeed.com/bufferedreader-vs-scanner/, [Diakses 15 Januari 2021].