LAPORAN TUGAS PROYEK

“Aplikasi Perhitungan Gaji Pegawai Dengan Java”



Ita Masita

1302018006

**TEKNIK INFORMATIKA dan UNIVERSITAS MUSLIM INDONESIA**

**2020**

**BAB I PENDAHULUAN**

1. **Latar Belakang**

Perkembangan ilmu pengetahuan dan teknologi saat ini sangatlah pesat. Kini banyak aplikasi pemrograman yang telah diciptakan dan berkembang untuk mempermudah berbagai pekerjaan manusia. Berbagai model aplikasi mulai dari yang sifatnya edukasi, informatif, hiburan, dan banyak lainnya yang dapat kita nikmati saat ini. Terutama aplikasi yang berbasic perhitungan. Dalam hal ini penulis mencoba membuat model sederhana untuk berbagai metoda perhitungan gaji pegawai dengan menggunakan Application Programming Interface (API).

API merupakan kumpulan dari pustaka class atau komponen atau library, yang sudah disediakan oleh Sistem Operasi, dimana berisi kumpulan perintah yang membentuk sebuah komponen yang sangat mempermudah para programmer dalam membangun aplikasi. Tentunya model aplikasi apapun dapat kita bangun menjadi aplikasi yang sangat variatif dan dapat digunakan dengan mudah.

Pada tugas kali ini penulis mencoba membangun aplikasi perhitungan sederhana untuk menghitunga total gaji seorang pegawai.

# Batasan Masalah

Untuk lebih mengarahkan pada uraian masalah agar tidak menyimpang dari pokok bahasan yang disajikan, maka penulis membatasi masalah pada hal-hal yang dianggap penting dan mendukung dalam pembuatan laporan ini. Pembahasan yang ada pada laporan ini mengenai pembuatan aplikasi perhitungan gaji pegawai dengan Netbeans.

# Sistematika Penulisan Laporan

Laporan ini dibuat dengan sistematika penulisan sebagai berikut:

BAB I PENDAHULUAN, Berisi latar belakang pembuatan aplikasi, tujuan dari pembuatan aplikasi, batasan masalah dan sistematika penulisan laporan.

BAB II LANDASAN TEORI, Berisi teori-teori mengenai software yang di pakai dan sebagai penunjang dalam pembuatan aplikasi ini.

BAB III ANALISIS, Menerangkan tentang spesifikasi hardware dan tool yang digunakan dalam pembuatan aplikasi.

BAB IV IMPLEMENTASI, Menerangkan tentang Syntaks dan tampilan akhir dari aplikasi yang dibuat.

# BAB II LANDASAN TEORI

* 1. **Konsep OOP**

Desain berorientasi object adalah sebuah teknik yang memfokuskan desain pada object dan class berdasarkan pada skenario dunia nyata. Pemrograman berorientasi objek diciptakan untuk mempermudah pengembangan program dengan cara mengikuti model yang telah ada dalam kehidupan nyata. Dalam paradigma ini, sesuai dengan model kehidupan nyata, segala bagian (entiti) dari suatu permasalahan adalah objek. Hal ini menegaskan keadaan (state), behaviour dan interaksi dari objek. Objek-objek ini kemudian juga dapat berupa gabungan dari beberapa objek yang lebih kecil. Sebagai contoh, tengoklah sebuah mobil. Mobil adalah sebuah objek dalam kehidupan nyata. Namun mobil sendiri terbentuk dari beberapa objek yang lebih kecil seperti roda ban, mesin, jok, dll. Mobil sebagai objek yang merupakan gabungan dari objek yang lebih kecil dibentuk dengan membentuk hubungan antara objek-objek penyusunnya. Begitu juga dengan sebuah program. Objek besar dapat dibentuk dengan menggabungkan beberapa objek-objek dalam bahasa pemrograman. Objek-objek tersebut berkomunikasi dengan saling mengirim pesan kepada objek lain. Selain itu juga menyediakan manfaat akan kebebasan pengembangan, meningkatkan kualitas, mempermudah pemeliharaan, mempertinggi kemampuan dalam modifikasi dan meningkatkan penggunaan kembali software.

Konsep-konsep pemrograman berorientasi objek dalam Java secara umum sama dengan yang digunakan oleh bahasa-bahasa lain. Jadi kebanyakan konsep yang kita bahas juga terdapat dalam bahasa selain Java. Namun, terkadang terdapat perbedaan-perbedaan kecil antara penerapan konsep-konsep tersebut dalam masing-masing bahasa. Perbedaan-perbedaan ini juga akan dijelaskan seiring penjelasan masing-masing konsep. Kelas merupakan inti dari pemograman java. Java itu sendiri merupakan pemograman yang mendukung dan mengimplementasikan konsep pemograman berorientasi objek sepenuhnya.

# Objek

Baik dalam dunia nyata atau dalam sebuah program, sebuah objek memiliki dua karakteristik, yaitu *state* dan *behaviour*. State adalah keadaan dari sebuah objek, seperti mobil memiliki state warna, model, tahun pembuatan, kondisi, dll. Sedang behaviour adalah kelakuan dari objek tersebut, seperti mobil dapat melaju, membelok, membunyikan klakson, dll. Objek

menyimpan statenya dalam satu atau lebih variabel, dan mengimplementasikan behaviournya dengan metode. Dengan penjelasan di atas, dapat disimpulkan bahwa objek adalah bagian software yang dibentuk dengan variabel-variabel dan metode-metode yang berhubungan dengan variabel tersebut.

Dengan karakteristik tersebut, kita dapat memodelkan berbagai objek dalam kehidupan nyata ke dalam objek-objek dalam sebuah program. Lebih lanjut kita dapat memodelkan objek- objek abstrak ke dalam sebuah program. Contoh umum untuk konsep abstrak seperti ini adalah objek Event, yaitu objek untuk mewakili peristiwa klik atau tombol ditekan.

Sebuah objek yang dibentuk dari sebuah kelas biasa disebut instans dalam terminologi OOP. Artinya objek tersebut adalah wujud nyata dari sebuah kelas. Variabel dan metode dari instans ini disebut *variabel instans* dan *metode instans*. Setiap instans menyimpan variabelnya sendiri-sendiri, jadi nilai variabel untuk tiap instans bisa berbeda.

# Kelas

Kelas adalah semacam cetakan, atau template, untuk membuat objek. Ibaratkan sebuah rancangan rumah yang digunakan untuk membangun ratusan rumah. Rumah yang dibangun tersebut adalah objek dari kelas rancangan rumah. Hal ini dapat dilakukan karena semua objek rumah yang dibangun memiliki karakteristik yang sama, sehingga dapat dibuatkan semacam blueprintnya. Tetapi objek-objek yang dibangun tetap akan memiliki bentuk fisik tertentu sendiri-sendiri, seperti variabel dalam sebuah program, atau pintu sebuah objek rumah. Dengan penjelasan ini, kelas dapat kita definisikan kembali menjadi sebuah blueprint, atau prototipe, yang mendefinisikan variabel dan metode yang sama untuk semua objek sejenis.

Sebagai contoh, misalkan kita ingin membuat kelas Rumah, maka kita harus membuat sebuah kelas yang mendefinisikan semua variabel yang dimiliki objek dari kelas tersebut. Selain itu, kelas tersebut juga harus mendeklarasikan metode-metode yang dimiliki objek dari kelas dan juga membuat implementasi dari metode tersebut. Dengan adanya kelas ini, kita dapat membuat sebanyak apapun objek-objek rumah yang sejenis, yaitu jenis yang didefinisikan oleh kelas Rumah. Setiap objek Rumah diciptakan, sistem akan mengalokasikan sejumlah memori untuk objek tersebut dan variabel-variabelnya. Dengan begitu setiap objek akan memiliki salinan masing-masing untuk setiap variabel instans.

Variabel kelas sebenarnya sama dengan variabel instans. Bedanya adalah, setiap objek berbagi satu dan hanya satu variabel kelas, tapi masing-masing memiliki salinan dari variabel

instans. Misalkan kelas Rumah yang kita buat hanya akan mendukung 2 lantai, dan setiap objek Rumah terdiri atas 2 lantai. Maka informasi ini cukup disimpan satu kali, karena nilainya tidak berbeda untuk semua objek. Lebih jauh, bila ada satu objek yang mengubah nilai dari variabel kelas, maka semua objek sejenis lainnya akan mengikuti perubahan itu. Di samping variabel, terdapat juga *metode kelas*. Metode jenis ini dapat langsung dipanggil melalui kelas dan bukan dari instans kelas tersebut.

# Method

Method merupakan suatu operasi berupa fungsi-fungsi yang dapat dikerjakan oleh suatu object. Method didefinisikan pada class akan dipanggil melalui object. Dengan kata lain method juga dikenal sebagai fungsi atau prosedur.

Pemanggilan method dalam suatu kelas dilakukan dengan menuliskan objek pemiliknya dan diikuti oleh operator titik (.) beserta nama method yang akan di eksekusi.

# Pewarisan Sifat Objek (Inheritance)

Pewarisan adalah salah satu cirri pemograman berorientasi objek, yang menyatakan suatu kelas dapat diturunkan lagi menjadi kelas-kelas baru yang lainnya sehingga dapat membentuk sebuah hiraki. Kelas yang merupakan kelas turunan ini bisa disebut dengan kelas anak (subclass) dan kelas yang menjadi dasar penurunan disebut kelas orang tua (superclass).

1. Dasar Pewarisan (Superclass dan Subclass)

Pemograman java mengizinkan kita untuk mendefinisikan suatu kelas yang bersifat generic. Selanjutnya kelas tersebut diturunkan lagi menjadi kelas baru yang bersifat lebih spesifik. Adapun kelas baru yang hasil turunan disebut subclass. Pada proses penurunan kelas ini kelas turunan akan mewarisi sifat-sifat yang terdapat pada kelas induknya, selanjutnya, kelas turunan tersebut dapat memiliki sifat-sifat spesifik yang sebelumnya tidak dimiliki oleh kelas induk. Sebagai contoh, yakni terdapat kelas binatang, kelas tersebut selanjutnya akan diturunkan lagi menjadi kelas-kelas baru yang lebih spesifik tanpa meninggalkan sifat-sifat dari kelas binatang, menjadi kelas: Herbivora (binatang pemakan tumbuhan) dan Karnivora (binatang pemakan tumbuhan). Kelas turunan yang lain akan diturunkan lagi menjadi kelas turunan yang lainnya seperti kelas herbivora bisa diturunkan lagi menjadi kelas Kambing dan Kerbau, sedangkan untuk kelas Karnivora dapat diturunkan lagi menjadi Anjing, Harimau, Singa dsb.

( ) kelas binatang adalah superclass dari kelas herbivoran dan karnivora itu sendiri berperan sebagai subclass. kemudian dari superclass tersebut terdapat subclass nya yakni pada superclass Herbivora dapat menjadi Subclass dari kelas Kambinga dan Kerbau. Begitupun pada superclass Karnivora dapat menjadi subclass dari kelas Harimau dan Kucing.

Pada pemograman java menyediakan kata kunci extends yang digunakan untuk proses penurunan terhadap suatu kelas. Bentuk umum dari penggunaan kata kunci tersebut adalah sebagai berikut:

Class nama-subclass extends nama-superclass{

// badan kelas

}

Ketika kita sedang membentuk sebuah hiraki kelas constructor yang akan dipanggil pertama kali dalam setiap pembentukan objek adalah constructor dari kelas induk paling dasar. Sebagai contoh kita memiliki induk dengan nama A dan diturunkan lagi menjadi kelas C, pada situasi saat ini setiap kita membentuk objek dari kelas C, maka urutan yang dipanggil adalah contructor dari kelas A disusul dengan contructor dari kelas B, baru kemudian contructor dari kelas c. Pada saat kita membentuk atau melakukan instansiasi terhadap suatu objek turunan, maka sebelumnya java juga akan membentuk objek dari kelas induk-induk nya.

1. Pengertian Polimorfisme

Polimorfisme adalah kemampuan suatu objek untuk mengungkap banyak hal melalui satu cara yang sama. Polimorfisme merupakan salah satu hal esensial dalam pemograman berorientasi objek karena alasan beriku: yakni polimorfisme mengizinkan kelas induk untuk mendefinisikan sebuah method general ( bersifat umum) untuk semua kelas turunannya. Polimorfisme dapat berupa overloading atau overriding.

Overloading merupakan bentuk polimorfisme yaitu beberapa metode yang dapat memiliki nama yang sama dengan isi dan parameter yang berbeda didalam sebuah kelas. Biasanya eksekusi program akan langsung mengacu pada metode yang dipanggil sesuai dengan parameter

Overriding merupakan bentuk polimorfisme yaitu berupa metode pada kelas orang tua yang dapat di tulis ulang pada kode kelas anak dalam pewarisan (inherentance) dengan memiliki nama yang sama dan memiliki isi ataupun parameter yang sama atau berbeda.

# Netbeans

NetBeans mengacu pada dua hal, yakni platform untuk pengembangan aplikasi desktop javadansebuah 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.

Application Programming Interface (API) adalah kumpulan dari pustaka class atau komponen atau library, yang sudah disediakan oleh Sistem Operasi, yang berupa kumpulan perintah yang membentuk sbuah komonen, sehingga akan membantu para programmer dalam membangun sebuah aplikasi.

# BAB III ANALISIS

Dalam pembuatan aplikasi kalkulator berjava ini, penulis mengunakan beberapa tools dan hardware sebagai pendukung dalam pembuatannya aplikasi ini. Tools yang digunakan adalah :

# Hardware

|  |  |
| --- | --- |
| Harddisk | 149.04 GB |
| RAM | 1016MB |
| Processor | Intel® Atom ™ CPU N270 |
| VGA | 128 MB |

**Tools**

*Table 3.1 Spesifikasi Hardware*

Dalam pembuatannya penulis menggunakan program Netbeans IDE 6.8.



*Gambar 3.1 Netbeans IDE 6.8*

# BAB IV IMPLEMENTASI

Bab ini akan menjelaskan implementasi program beserta tampilan – tampilan form pada aplikasi yang telah diselesaikan.

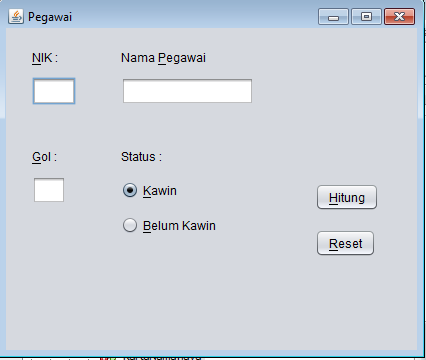
Berikut permasalahannya :

|  |
| --- |
| Pegawai |
| Sting NIK; String Nama; integer Golongan  integer Status |
| + Set/Get |

Tunjangan = 10% dari Gaji Toatal = Gaji + Tunjangan

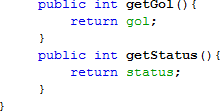
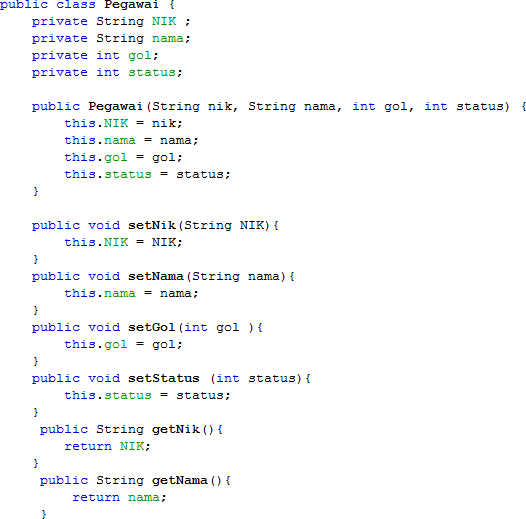
|  |
| --- |
| Proses |
| + Proses  + Hitung  + Tampil |

Dan frameform yang akan dibuat adalah frame Pegawai.

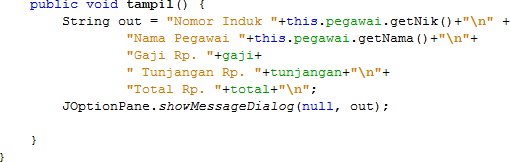
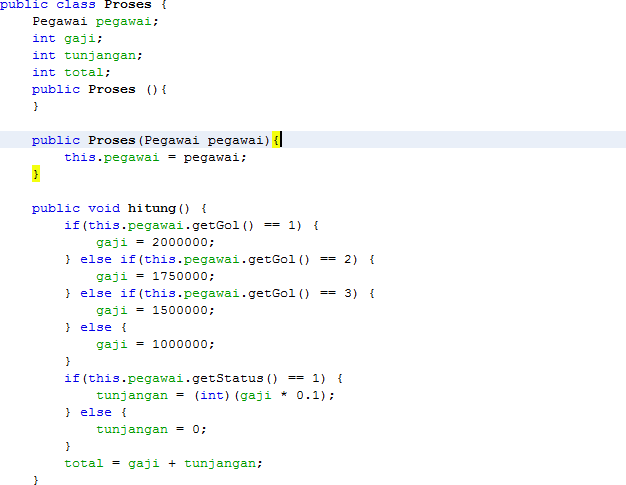


Sebelum membuat framenya, kita harus membuat kelas terlebih dahulu, dan dalam percobaan ini dibuat class Pegawai () dan class Proses () yang merupakan turunan dari class Pegawai.

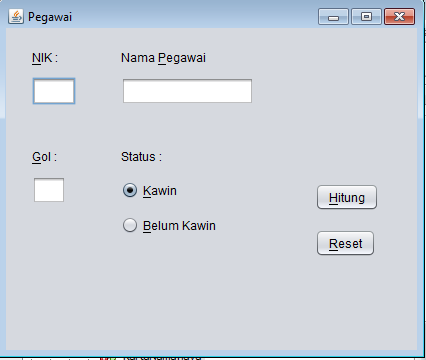
# Class Pegawai



**Class Proses**



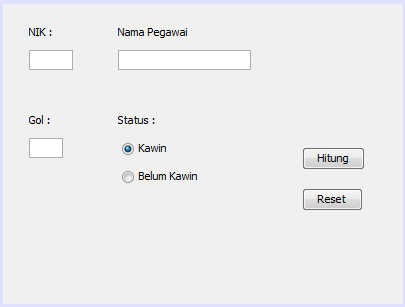
Selanjutnya membuat desain untuk frame Pegawai, fungsi yang dibutuhkan adalah 4 classjLabel, 2 jTextField, 1 jComboBox, 2 jRadioButton, dan 2 komponen jButton.



Lalu mulailah dengan mendesain Framenya sesuai dengan table dibawah ini:

|  |  |  |
| --- | --- | --- |
| Class | Properties | Values |
| jLabel1 | Text | NIK |
|  | labelFor | jTextField1 |
| jLabel2 | Text | Nama Pegawai |
|  | labelFor | jTextField2 |
| jLabel3 | Text | Gol |
|  | labelFor | jTextField3 |
| jLabel4 | Text | Status |
|  | labelFor | jRadioButton |
| jTextField1 | Text |  |
| jTextField2 | Text | Pegawai |
| jComboBox1 | Model | 1,2, 3, 4 |
| jRadioButton1 | Text | Kawin |
|  | Mnemonic | K |
| jRadioButton2 | Text | Belum Kawin |
|  | Mnemonic | P |
| jButton1 | Text | Hitung |
|  | Mnemonic | H |
|  | Tooltoptext | Menghitung Gaji, Tunjangan  dan Total |
| jButton2 | Text | Reset |
|  | Mnemonic | R |
|  | Tooltoptext | Hapus dan Ulang |

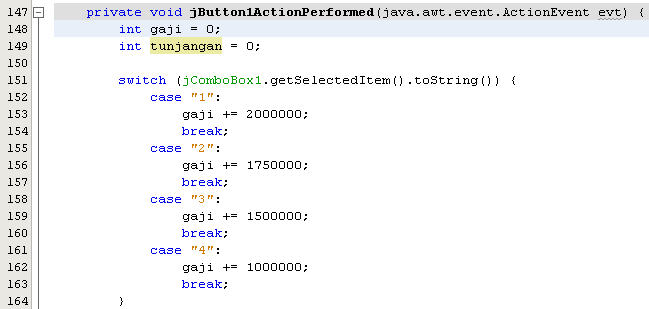
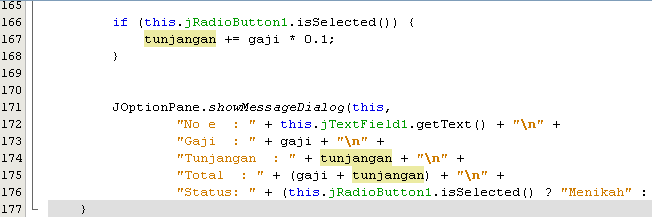
Desain Framenya



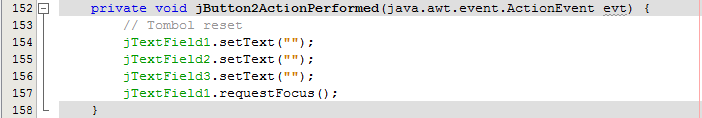
Untuk menjelaskan beberapa bagian dari tiap form beserta kode programnya, kita harus memanggil kelas Proses () yang sudah menjadi turunan dari class Pegawai () terlebih dahulu dengan menulis kode program:

C:\Users\user\Documents\tugas 4 java\6.png

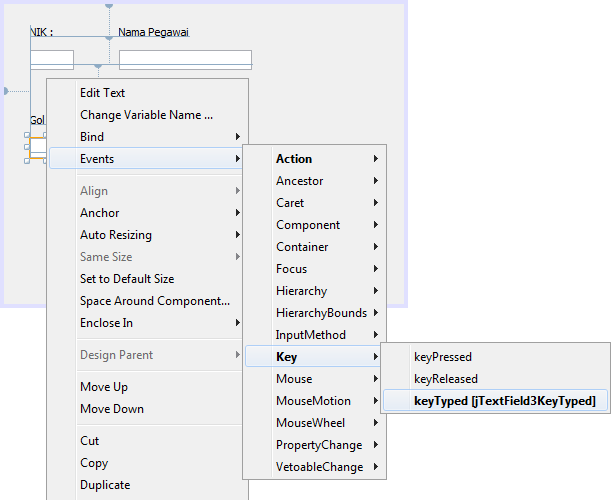
Untuk meng-klik tombol Hitung, maka akan mengeksekusi besar gaji dari golongan terlebih dahulu, lalu menghitung tunjangan yang di dapat 10% dari besar gaji, dan total dihitung dari jumlah tunjangan dan gaji. Berarti kita harus menuliskan kode program pada tombol Hitung. Klik 2 kali pada tombol Hitung, maka akan berpindah ke layar editor, tulis kode programnya:



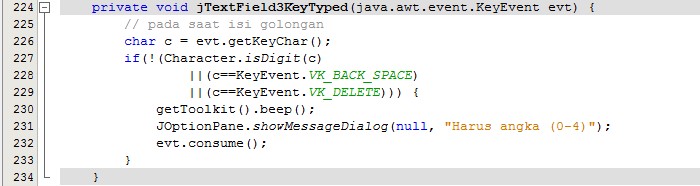
Jika meng-klik tombol reset. Berarti kita harus menuliskan kode program pada tombol ini. Klik 2 kali pada tombol Reset, maka akan berpindah ke layar editor, tulis kode programnya:



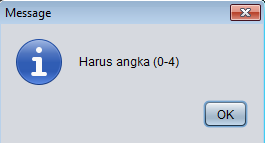
Pada saat mengisi pada kolom Golongan masih belum diatur variabelnya, maka kita akan mengunci type text agar difokuskan kepada integer, langkahnya :



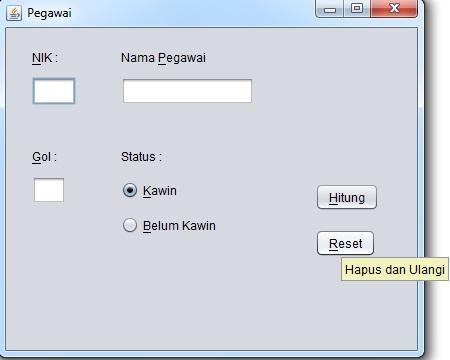
maka akan berpindah ke layar editor, tulis kode programnya:



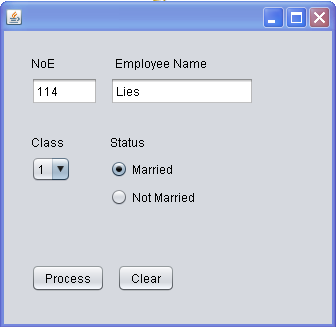
Maka hasilnya setelah di cek akan muncul peringatan seperti ini :



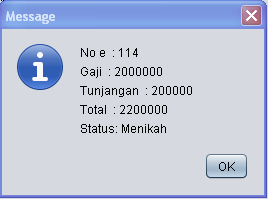
Hasil frame setelah menggunakan Tooltoptext:



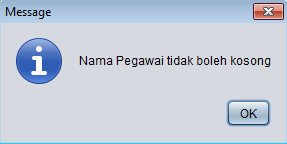
Hasil Running setelah menggunakan Rumus :



Menghasilkan output seperti dibawah ini :



Ketika Nama Pegawai tidak di masukkan, akan muncul message seperti ini :



KESIMPULAN

Aplikasi ini saya buat untung menghitung gaji kariawan agar mempermudah para bisnismen,dengan menggunakan aplikasi Netbeans

SARAN

Meskipun penulis menginginkan kesempurnaan dalam penyusunan laporan ini akan tetapi pada kenyataannya masih banyak kekurangan yang perlu penulis perbaiki. Hal ini dikarenakan masih minimnya pengetahuan penulis. Oleh karena itu kritik dan saran yang membangun dari para pembaca sangat penulis harapkan sebagai bahan evaluasi untuk kedepannya.

DAFTAR PUSTAKA

https://www.petanikode.com/java-oop/