GeRuDok

Smjernice za programiranje

Verzija 1.0

Istorija revizija

|  |  |  |  |
| --- | --- | --- | --- |
| **Datum** | **Verzija** | **Opis** | **Autor** |
| 10/11/2016 | 1.0 | Početna verzija standarda kodiranja | Miloš Nikolić |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

Sadržaj

1. Uvod 4

1.1 Svrha 4

1.2 Područje 4

1.3 Definicije, skraćenice I akronimi 4

1.4 Reference 4

1.5 Pregled 4

2. Stil i organizacija koda 4

3. Komentari 4

3.1 Klase 4

3.2 Funkcije 4

4. Dodjeljivanje imena 4

4.1 Imena klasa 5

4.2 Imena namespace-a 5

4.3 Imena funkcija 5

4.4 Imena promjenjivih 5

4.5 Imena konstanti 5

5. Deklaracije 5

6. Izrazi i naredbe 5

7. Rad sa memorijom 5

8. Rukovanje greškama i izuzecima 5

9. Prenosivost 5

10. Ponovna upotreba 5

11. Problemi kompilacije 6

12. Sažeti standard 6

Smjernice za programiranje

# Uvod

Ovaj dokument opisuje standarde i kovencije koje se tiču programiranja, tj. opisuje način kako se piše programski kod.

## Svrha

Svrha ovoga dokumetna je da opiše standarde i konvencije koje se koriste prilikom pisanja programskog koda.

## Područje

Ovaj dokument obuhvata čitav projekat “GeRuDok”

## Definicije, skraćenice i akronim

Pogledati Pojmovnik.docx.

## Reference

[This subsection provides a complete list of all documents referenced elsewhere in the **Programming Guidelines**. Identify each document by title, report number if applicable, date, and publishing organization. Specify the sources from which the references can be obtained. This information may be provided by reference to an appendix or to another document.]

## Pregled

Nastavak dokumenta sadrži definicije elemenata koji će biti korišteni u kodiranju i način na koji će biti korišteni.

# Stil i organizacija koda

Kod mora da bude “čist” i rasporedjen u odgovrajuće pakete. Obavezujuće je i komentarisanje koda kako bi programeri bili bolje upućeni u samu logiku. Kao jedincu za uvlačenje blokova naredbi koristiti jedan tab. Linije koga ne smiju biti duže od

# Komentari

Prilikom kodiranja, mogu se koristiti dvije vrste komentara:

* Komentari koji počinju sa “/\*”, a završavaju sa “\*/”, koriste se kod komentara dužih od jedne linije
* Inline komentari koji počinju sa “//” i koriste se za kraće komentare

Komentare pisati na srpskom jeziku.

## Klase

Komentarisanje klasa mora sadržati dokumentaciju u kojem se navodi ime autora, ime klase, njenu svrhu kao i datum kreiranja te klase. Ovo mora biti urađeno u DOC formatu. Ukoliko se referenciraju druge klase, potrebno je postaviti link prema tim klasama.

## Funkcije

Komentari funkcija moraju sadržati naziv funkcije, kratki opis, parametre koje prima kao i vrijednost koju vraća (ako vraća vrijednost).

# Dodjeljivanje imena

Imena svih klasa i promjenjivih moraju biti na srpskom jeziku. Imena moraju biti smišljena i što manje dužine.

## Imena klasa

Nazivi klasa se pišu u Camel-Casing stilu sa time što je prvo slovo imena klase veliko.

## Imena namespace-a

Imena namespace-a se pišu malim slovima.

## Imena funkcija

Imena funkcija se pišu u Camel-Casing stilu sa time što je prvo slovo imena funckije malo

## Imena promjenjivih

Imena promjenjivih se pišu u Camel-Casing stilu sa time što je prvo slovo imena promjenljivih malo.

## Imena konstanti

Imena konstanti pišu se velikim slovima sa time što se riječi kod konstanti od više riječi odvajaju \_(underscore) znakom.

# Deklaracije

Lokalne promjenjive se deklarišu pri vrhu bloka koda, odnosno na početku posle vitičaste zagrade. U jednoj liniji se nalazi deklaracija jedne lokalne promjenjive ne više promjenjivih.

Komponente striktno deklarisati kao public promjenljive unutar klase Frame-a.

# Izrazi i naredbe

Kada izraz ne može stati u jedan red, postupa se prema sljedećim pravilima:

* Prelazak u novi red nakon zareza.
* Prelazak u novi red nakon operatora.
* Poravnanje nove linije sa početkom izraza na istom nivou kao prethodna linija.
* Pri deklaraciji izraza za kontrolu toka programa uvijek koristiti vitičaste ({, }) zagrade bez obzira da li se radilo o jednoj komandi ili ne.
* Između aritmetičkih i logičkih znakova uvijek praviti tačno jedan razmak.
* Sve izraze unutar vitičaste ({, }) zagrade uvući za jedan tab.

# Rad sa memorijom

Java programski jezik ima ugrađen “Garbage Collector” koji upravlja i vodi računa o memoriji tako da programer ne mora da vodi računa o tome. Ako se slučajno desi da objekat nije opslužen od strane Garbage Collectora, tada programmer mora eksplicitno da uništi objekat koristeći destructor.

# Rukovanje greškama i izuzecima

Izuzeci se koriste isključivo za rukovanje logičkim i programskim greškama, konfiguracionim greškama, oštećenim podacima i iscrpljenim resursima. Sve naredbe za koje se smatra da su potencijalno rizične za rad i stabilnost sistema je potrebno uokviriti u try/catch blok. Kada se u radu aplikacije javi neka greška, potrebno je na određeni način obavijestiti korisnika da je došlo do greške i na koji način će grešku otkloniti i nastaviti sa radom.

# Prenosivost

GeRuDok je zbog portabilnosti Java koda, podržan od strane Windows OS, kao i većine distribucija Linux operativnog sistema.

# Ponovna upotreba

Kod treba da bude u potpunosti modularan i pravilno podijeljen po paketima kako bi se kasnije mogao lako ponovo iskoristiti.

# Problemi kompilacije

[This section describes dependencies between modules, compiler directives, and so on.]

# Sažeti standard

* Komentare pisati na srpskom jeziku.
* Imena moraju biti smišljena u Camel-Casing stilu na sprskom jeziku.
* Ne pisati vise komandi u jednoj liniji.
* Sve izraze unutar vitičaste ({, }) zagrade uvući za jedan tab.
* Koristiti try-catch za kritični dio koda.
* Komponente striktno deklarisati kao public promjenljive unutar klase Frame-a.