**

queri

Smjernice za programiranje

Verzija 1.0

Istorija revizija

|  |  |  |  |
| --- | --- | --- | --- |
| **Datum** | **Verzija** | **Opis** | **Autor** |
| 03.05.2022. | 0.1 | Napisan uvod. | Milan Vlaški |
| 03.05.2022. | 0.1.1 | Ispravljene greške. Pravilno komitovano. | Milan Vlaški |
| 05.05.2022. | 0.1.2 | Dodat još jedan izvor. | Milan Vlaški |
| 05.05.2022. | 0.2 | Urađeno do imenovanja. (DOVRŠITI TABELU) | Milan Vlaški |
| 06.05.2022. | 0.2.1 | Napisana deklaracija | Milan Todorović |
| 06.05.2022. | 0.2.2. | Napisani izrazi i izjave | Milan Todorović |
| 07.05.2022. | 0.3 | Dovrsena tabela u "imenovanje". | Milan Vlaški |
| 09.05.2022. | 0.4 | Mala dorada. | Milan Vlaški |
| 09.05.2022 | 0.5 | Dodate 7,8 i 9 stavka | Milan Todorović |
| 10.05.2022. | 0.6 | Dodate 10, 11, 12 stavka i dorađene prethodne. | Milan Vlaški |
| 11.05.2022. | 0.7 | Popunjene stavke koje su falile. | Milan Vlaški |
| 17.05.2022. | 1.0 | Dorada i kompletiranje dokumenta | Mladen Todorović |

Sadržaj

1. Uvod 4

1.1 Svrha 4

1.2 Područje primjene 4

1.3 Definicije, akronimi i skraćenice 4

1.4 Reference 4

1.5 Pregled 4

2. Organizacija i stil koda 4

2.1 Indentacija (uvlačenje) 4

2.2 Dužina linije 4

2.3 Razbijanje redova 4

3. Komentari 5

3.1 Komentari dokumentacije 5

3.2 Komentari implementacije 5

4. Imenovanje 5

5. Deklaracije 6

6. Izrazi i izjave 6

6.1 Izrazi 6

6.2 Izjave 6

6.2.1 Return izjave 7

6.2.2 If/else izjave 7

6.2.3 For izjave 7

6.2.4 While izjave 7

6.2.5 Do-while izjava 7

6.2.6 Switch izjava 8

6.2.7 Try-catch izjava 8

7. Upravljanje memorijom 8

8. Upravljanje greškama i izuzecima 8

9. Prenosivost 8

10. Ponovno korištenje 8

11. Problemi pri kompajliranju 9

12. Aneks: sažetak smjernica 9

Smjernice za programiranje

# Uvod

Ovaj dokument sadrži detaljno opisane smjernice za programiranje tj. pisanje Java koda, i namijenjen je za programere koji rade na razvoju softverskog alata *queri*.

## Svrha

Svrha dokumenta je da izloži konvencije programiranja kojima će se voditi programeri koji rade na projektu. Ovim se izbjegavaju sukobi stilova i omogućava se jasan i čitljiv kod.

## Područje primjene

Dokument utiče na način kodiranja prototipa i finalnih verzija softverskog alata *queri.* Samim tim ograničava realizaciju na okruženje Eclipse, u programskom jeziku Java uz dodatak Swing seta alata za GUI.

## Definicije, akronimi i skraćenice

Sve definicije, akronimi i skraćenice, korištene u dokumentu, su opisane u dokumentu Rječnik, koji je dio projektne dokumentacije.

## Reference

1. Code Conventions for the Java Programming Language (<https://www.oracle.com/java/technologies/javase/codeconventions-introduction.html>)
2. Google Java Style Guide (<https://google.github.io/styleguide/javaguide.html>)
3. Stackify blog (<https://stackify.com/best-practices-exceptions-java>)

## Pregled

U nastavku dokumenta Smjernice za programiranje opisana je organizacija i stil koda kao i način pisanja komentara, imenovanja, deklaracija, izraza i izjava, upravljanja memorijom, upravljanja greškama i izuzecima, opis prenosivosti, ponovnog korištenja i potencijalnih problema tokom kompajliranja.

# Organizacija i stil koda

## Indentacija (uvlačenje)

Kao jedinicu uvlačenja treba koristiti 4 razmaka. Eclipse ima ugrađen formater koji se brine o pravilnom uvlačenju pojedinih sekcija koda.

## Dužina linije

Dužina linije treba biti manja od 80 karaktera.

## Razbijanje redova

Kada red ne može ispoštovati gornje pravilo, pisati u novi, vodeći se sljedećim pravilima:

* Razbiti poslije zareza
* Razbiti prije operatora
* Novu liniju poravnati sa početkom izraza u prošloj liniji
* Poželjno je razbijati na višem sintaksičkom nivou (nakon zatvaranja zagrade, “.” separatora itd) .

Ako je kod još uvijek nejasan, onda ga uvući za 8 redova.

# Komentari

Pisanje komentara uz kod predstavlja dobru praksu. Iako se nastoji da sam kod bude pregledan i lako razumljiv, određeni dijelovi koda su pogodni za dodatno pojašnjenje. Ta pojašnjenja mogu biti “zlata vrijedna” kada različiti ljudi vrše određene izmjene u kodu, ali i prilikom održavanja cijelog softvera poslije nekog dužeg vremena.

## Komentari dokumentacije

Netrivijalne metode, klase, interfejse opisati komentarima dokumentacije. Opisati jasno ulaz izlaz (gdje je to primjenljivo), srpskim jezikom. Okruženje Eclipse omogućava generisanje JavaDoc (Java dokumentacije). Počinju sa “/\*\*” i završavaju se sa “\*/”.

## Komentari implementacije

Komentare implementacije koristiti da bi svrha koda bila jasna na prvi pogled, ili da pruži informaciju koja nije prisutna ili lako razumljiva u samom kodu. Komentare je potrebno na odgovarajući način odvojiti od samog koda. Ne koristiti kosu crtu unutar komentara implementacije. U Eclipse okruženju razlikujemo:

* jednolinijske komentare: počinju sa “//” i idu do kraja linije koda,
* višelinijske komentare počinju sa “/\*” i završavaju se sa “\*/”.

# Imenovanje

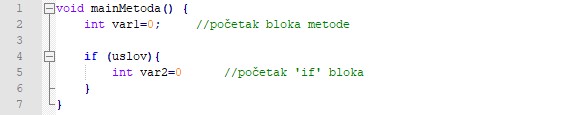
| **Tip identifikatora** | **Pravila imenovanja** | **Primjeri** |
| --- | --- | --- |
| Paketi | Prefiks unikatnog imena paketa se uvijek piše svim malim ASCII slovima. Koriste se obrnuti domeni. U slucaju da postoje ključne riječi sadržane u imenu domene, ili ime počinje brojem, tada se ispred ubacuje znak “ ”. | com.sun.eng  com.apple.quicktime.v2  edu.cmu.cs.bovik.cheese |
| Klase | Klasna imena trebaju biti imenice, sa prvim slovom svake riječi velikim. Koristiti čitave riječi, a ne skraćenice (osim ako su skraćenice poznatije kao takve npr. URL ili HTML). Svaka sljedeća riječ počinje velikim slovom. | class Raster; class ImageSprite; |
| Interfejsi | Imena interfejsa treba pisati kao i imena klasa. | interface RasterDelegate; interface Storing; |
| Metode | Metode moraju biti glagoli, sa malim početnim slovom prve riječi, a sa velikim početnim slovom ostalih riječi. | run(); runFast(); getBackground(); |
| Promjenljive | Promjenljive pisati sa istim pravilima kao i metode.  Imena trebaju biti kratka ali značajna, tj. ono što promjenljiva predstavlja treba biti jasno na prvi pogled. Imena dužine jednog karaktera izbjegavati, osim kod privremenih promjenljivih (npr. brojača). | int i;  char c;  float myWidth; |
| Konstante | Imena promjenljivih deklarisanih kao klasne konstante pišu se sa svim velikim slovima i riječima odvojenim donjom crtom (“\_”). | static final int MIN\_WIDTH = 4;  static final int MAX\_WIDTH = 999;  static final int GET\_THE\_CPU = 1; |

# Deklaracije

Jedna deklaracija po liniji radi lakšeg pisanja komentara.



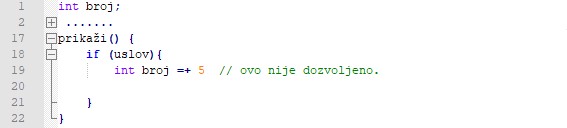
Deklaracija se piše na početku bloka koda.



Jedini izuzetak ovog pravila je za for petlju.



Zabranjene su lokalne deklaracije koje skrivaju globalne.



Deklaracija klasa i interfejsa se započinje otvorenom vitičastom zagradom “{“ odmah poslije njene deklaracije. Zatvorena vitičasta zagrada “}” stoji sama na kraju klase, osim ako je tijelo prazno, onda se piše odmah poslije otvorene vitičaste zagrade “{“.



# Izrazi i izjave

Izrazi izvršavaju operacije nad podacima i premiještaju podatke. Izjave predstavljaju osnovnu jedinicu izvršenja.

## Izrazi

U izrazima se mora voditi računa da ne bude dvosmiselnih dodjela, tj. da ne dolazi do eksplicitne konverzije podataka. Ukoliko se radi o složenim izrazima potrebno je korisiti zagrade radi lakšeg razumijevanja koda.



## Izjave

Svaka linija može imati najviše jednu izjavu.

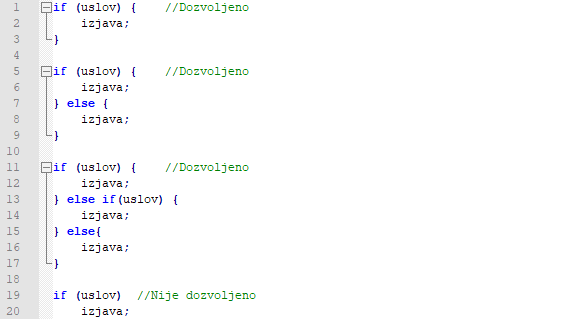


### Return izjave

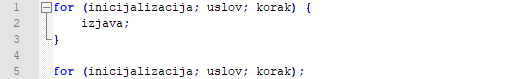
Return izjave ne bi trebale da se pišu sa zagradama, osim ako se radi o složenim izrazima.



### If/else izjave



### For izjave



Kada imamo više od tri izjave u inicijalizaciji ili u koraku, onda je potrebno inicijalizovati varijable prije for petlje ili na kraju petlje.

### While izjave

While izjava ima sljedeću formu:



Prazna while izjava:



### Do-while izjava

Do-while izjava ima sljedeću formu:



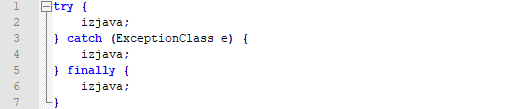
### Switch izjava

Switch izjava ima sljedeću formu:



### Try-catch izjava

Try-catch izjava ima sljedeću formu:



# Upravljanje memorijom

Koristi se sistem za automatsko upravljanje memorijom koji se naziva *garbage collector*, koji radi u pozadini i briše neiskorišćene objekte.

# Upravljanje greškama i izuzecima

* Try i catch blok se završava sa finally blokom.
* Prilikom kreiranja novog tipa izuzetka potrebno ga je dokumentovati.
* Potrebno je čuvati svaku grešku i izuzetak u log file.
* Prvo se hvataju specifični, pa uopšteni izuzeci.
* Prilikom bacanja izuzetaka potrebno je opisati razlog nastajanja greške korisniku.
* Ukoliko se desi izuzetak potrebno je omogućiti nesmetan rad korisnika.

# Prenosivost

Grafički editor *queri* može da radi na bilo kom računarskom sistemu koji posjeduje JVM. Izvršni kod se ne mora mijenjati da bi zadovoljio posebne potrebe računarskog sistema.

# Ponovno korištenje

* Pridržavati se principa modularnosti.
* Moduli moraju biti slabo spregnuti.
* Klase i metode trebaju da obavljaju jednu funkciju.
* Koristiti enkapsulaciju.
* Praviti testove za klase i učiniti da su klase jednostavne za testiranje.
* Ne ponavljati poslovni kod, jer je moguće da će biti mijenjan i nekompatibilan s ostatkom koda.

# Problemi pri kompajliranju

Potrebno je posjedovati sve potrebno za kompajliranje Java koda što je opisano u dijelu prenosivost. Koristiti alate za upravljanje zavisnostima između modula (Maven je preporučen). Obrisati neiskorišćene zavisnosti i komentarisati importe zavisnosti. Koristiti poruke u kodu u slučaju nastanka greški pri kompaljiranju.

# Aneks: sažetak smjernica

* Organizacija i stil koda opisuje izgled i formatiranje koda radi lakše čitljivosti.
* Komentari – opisano kada i kako komentarisati i dokumentovati kod.
* Imenovanje – data tabela s pravilima imenovanja elemenata kojima se programer služi.
* Deklaracije – opisan način deklarisanja promjenljivih.
* Izrazi i izjave – opisan način pisanja petlji, matematičkih i logičkih izraza programskog koda.
* Upravljanje memorijom – opisan rad garbage collectora.
* Upravljanje greškama i izuzecima govori o rješavanju mogućih greški pomoću try-catch izraza.
* Prenosivost opisuje uslove i mogućnosti prenosa queri na druge platforme.
* Ponovno korištenje – opisani principi dobre prakse koji omogućuju da kod bude ponovo iskoristiv u budućim projektima ili interno.
* Problemi pri kompajliranju – opisane mođusobne zavisnosti u cilju izbjegavanja problema pri kompajliranju.