

MakeitTrue - MiT

Smjernice za dizajn

Verzija 1.0

Istorija revizija

|  |  |  |  |
| --- | --- | --- | --- |
| **Datum** | **Verzija** | **Opis** | **Autor** |
| 10.11.2021. | 0.1 | Opšta podešavanja dokumenta. | Stefan Jokić |
| 11.11.2021. | 1.0 | Finalna verzija dokumenta. | Stefan Jokić |
| 12.11.2021. | 1.0 | Formatiranje dokumenta. | Milica Ivković |

Sadržaj

1. Uvod 4

1.1 Svrha 4

1.2 Namjena dokumenta 4

1.3 Pojmovi, skraćenice i akronimi 4

1.4 Reference 4

1.5 Pregled 4

2. Generalni standardi dizajna i implementacije 4

2.1 Mapiranje od dizajna do implementacije 4

2.2 Specifikacija interfejsa za podsisteme 4

2.3 Dokumentovanje operacija 4

2.4 Dokumentovanje poruka 5

2.5 Detektovanje, opsluživanje i izvještaji o greškama 5

2.6 Upravljanje memorijom 5

2.7 Distribucija softvera 5

2.8 Ponovna iskoristivost komponenti 5

2.9 Upotreba već kreiranih biblioteka 5

2.10 Ponovna iskoristivost komponenti 5

2.11 Modifikacija sistema i smjernice za razvoj 5

3. Smjernice dizajna arhitekture 5

Smjernice za dizajn

# Uvod

Dokument Smjernice za dizajn jasno definišu sve smjernice i služi za kreiranje podjele specifikacija dizajna kako bi se omogućilo da se izvrši bolja implementacija editora.

## Svrha

Svrha dokumenta Smjenice za dizajn je specificiranje detaljnog opisa standarda za dizajn, konvencija i idioma koji se koriste u dizajnu editora.

## Namjena dokumenta

Dokument Smjernice za dizajn obuhvata precizne smjernice vezane za projektovanje softvera, odnosno razvoj dizajna aplikacije.

## Pojmovi, skraćenice i akronimi

Svi korišteni, nepoznati pojmovi, akronimi i skraćenice su opisani u dokumentu Rječnik.

## Reference

Reference na dokument:

* Dokument Rječnik.

## Pregled

Dokument Smjernice za dizajn možemo posmatrati iz dva dijela. U prvom dijelu opisuje se sam sadržaj dokumenta, dok se u drugom dijelu nalazi detaljniji opis svih stavki dizajn sistema. Radi bolje preglednosti stavke su grupisane u odgovarajuće grupe.

# Generalni standardi dizajna i implementacije

## ****Mapiranje od dizajna do implementacije****

Do dizajna došlo se sekvencijalnim postupkom tako što se u analizi zahtjeva prvo kreirao model zahtjeva. Na osnovu modela zahtjeva kreiran je model slučaja korištenja preslikavanjem funkcionalnih zahtjeva u više slučajeva korištenja. Klasni dijagram je kreiran prema modelu slučajeva korištenja implementiranjem jednog slučaja korištenja pomoću jedne ili više klasa. Pri razvijanju aplikacije korištena je MVC arhitektura u cilju dobijanja što bolje struktuiranog modela. MVC model se može lako nadograditi, proširiti i lako je razumljiv sa programerske strane. Modelovanje se vrši pomoću softverskog alata SAP PowerDesigner, koji omogućava kreiranje osnovnih paketa MVC modela, te detaljno specificiranje potrebnih klasa, interfejsa i modela, kao i specificiranje veza među njima.

## ****Specifikacija interfejsa za podsisteme****

U paketu MVC modela view sadrži izgled aplikacije specificiraju se interfejsi različitih podstitema koji će u cjelini predstavljati standardizovani izgled grafičkog editora za kombinatorne mreže (MiT editora).

## ****Dokumentovanje operacija****

Programski kod MiT editora treba da bude dokumentovan. Pored programskog koda potrebno je dokumentovati sve slučajeve korištenja, sve klase, sve metode i sve promjenljive koje se koriste prilikom implementacije. Na ovaj način omogućeno je olakšano održavanje i proširenje editora. Prilikom implementacije koristiće se sljedeća konvencija:

* Paketi

Imena paketa se uvijek pišu malim ASCII slovima i trebaju biti na vrhu domenskih imena.

* Klase

Imena klasa su imenice, pri čemu se svaka riječ piše velikim slovom. Imena klasa trebaju biti kratka i deskriptivna. Svaka klasa mora da sadrži njen opis, autora, verziju i naziv klase.

* Interfejsi

Imena interfejsa poštuju pravila imenovanja kao što je slučaj i sa klasama.

* Metode

Imena metoda su glagoli, tako da je prvo slovo metode malo, a zatim prvo slovo svake riječi je veliko slovo.

* Promjenljive

Imena varijabli pišu se početnim malim slovom, a zatim prvo slovo svake riječi je veliko slovo.

* Konstante

Imena konstanti trebaju da se pisu velikim ASCII slovima. Za odvajanje riječi koristi se donja crtica (“\_”).

## ****Dokumentovanje poruka****

Parametri poruka neće biti dokumentovani, već samo konkretne poruke i kontekst u kojem se javljaju.

## ****Detektovanje, opsluživanje i izvještaji o greškama****

Za upravljanje izuzecima i greškama koristi se mehanizam izuzetaka. Eventualno, za neke visoko rizične dijelove sistema, može se koristiti parameter, koji će imati različite vrijednosti u odnosu da li je neka operacija uspješno ili neuspješno završena, i samim tim se koristiti za upravljanje greškama. Za programiranje koristi se programski jezik Java koji ima mehanizme kao što su try-catch blokovi.

## ****Upravljanje memorijom****

Za upravljanje memorijom koristiće se mehanizam Java programskog jezika, odnosno *Garbage collector*.

## ****Distribucija softvera****

Softver se distribuira preko internet, preuzimanjem sa zvanične web prezentacije MiT editora.

## ****Ponovna iskoristivost komponenti****

Pojedine komponente mogu ponovo iskoristiti u svrhu kreiranja editora različitih namjena.

## ****Upotreba već kreiranih biblioteka****

Pri implementaciji softvera koristiće se određene potrebne biblioteke Java programskog jezika.

## ****Ponovna iskoristivost komponenti****

Pojedine komponente radiće se generički kako bi se mogle iskoristiti pri kreiranju editora različitih namjena.

## ****Modifikacija sistema i smjernice za razvoj****

Za razvoj editora koristi se programski jezik Java. Iz tih razloga editor ne zahtjeva dodatne usluge od sistema ili određene hardverske dodatke. Da bi editor funkcionisao potrebno je posjedovati Java virtuelnu mašinu.

# Smjernice dizajna arhitekture

Arhitektura softvera mora biti podijeljena na module, tako da pri odstranjivanju jednog modula ostatak sistema može nesmetano da funkcioniše. Softver treba da obezbjedi nesmetanu zamjenu dijelova sistema boljim verzijama. Arhitektura softvera se temelji na MVC šablonu.