Specifikacija programskog koda

Elektroenergetski softverski inžinjering

6/2/2014

Sadržaj

Uvod

U okviru stanadardizacije nastavnih materijala na ESI smeru predviđeno je da programski kod i prateća dokumentacija (komentari) koji studenti dobiju od profesora i asistenata budu međusobno poravnati po pitanju stila.

Svrha ovog dokumenta je predlog definicije skupa pravila za pisanje i dokumentovanje programskog koda. Ova pravila se odnose na sav kod koji studenti dobijaju od profesora i asistenata kao i programski kod koji razvijaju u okviru vežbi. Stil pisanja bi takođe trebalo uzeti u obzir prilikom ocenjivanja studentskih radova.

Dokument je podeljen na tri celine:

- Opšta pravila koja se odnose na C/C++ i C# programske jezike,
- Pravila specifična za C# programski jezik.
- Pravila specifična za C/C++ programske jezike,

U ovom dokumentu neće biti pokriveno pravilno pisanje koda sa stanovišta optimalnog izvršavanja već čitljivosti. Kao osnova za dokument je iskorišćen [1] koji je preveden, uprošćen i dopunjen pravilma za C/C++ programski jezik.

Opšta pravila

1. Uvek koristiti kamilju ili paskal notaciju:
   - ovoJeKamiljaNotacija,
   - OvoJePaskalNotacija.
2. Ne koristiti mađarsku notaciju.
3. Izbegavati nazive cele napisane velikim slovima sem u slučaju predprocesorskih direktiva u C/C++ jezicima.
4. Ne koristiti imena koja se razlikuju od prethodnih definicija samo po veličini slova:
   - `class ReplicationMessage { ... }`
   - `class Replicationmessage { ... }`
5. Ne počinjati imena brojem.
6. Dozvoljeno je dodavati numeričke sufkse na kraju imena ukoliko imaju smisla.
7. Imena moraju biti razumljiva, imati značenje i odgovarati kontekstu.
8. Težiti pisanju razumljivog a ne sažetog programskog koda.
9. Izbegavati korišćenje skraćenica sem kada je puno ime suviše dugačko.
10. Izbegavati skraćenice duže od 5 karaktera.
11. Koristiti velika slova za skraćenice od dva slova a paskal notaciju za duže.
12. Svaka skraćenica mora biti poznata ili dokumentovana.
13. Ne koristiti rezervisane reči kao imena.
14. Izbegavati konflikte sa postojećim domenima imena (name spaces) i tipovima,
15. Ne dodavati redundantne ili besmislene prefikse idenfikatorima:
   - `class CSemaphore { ... }`
   - `struct SampleStructure { ... }`
16. U propertima i podacima članovima klase ne uključivati ime klase kojoj pripadaju:
```
class Message
{
    int MessageType { get; set; }
}
```
17. Pokušati bool promenljivama dodeliti prefix: “Can”, “Is”, “Has”:
   - `bool IsValid;`
18. Dodati smislene prefikse ili sufikse imenima kao: Average, Min, Max, Count:
   - `AverageSum, MinVelocity...`
19. Funkcije koje vraćaju `bool` ili promenljive tipa `bool` u uslovnim iskazima ne porediti sa `true` ili `false` već ispitivati njihovu povradnu vrednost, odnosno vrednost u slučaju promenljive.
20. Ne koristiti `goto`.
21. Kompletan programski kod i komentari trebaju biti na engleskom jeziku.

Format programskog koda

1. Ne definisati više od jednog domena imena po fajlu.
2. Izbegavati stavljanje više klasa u jedan fajl.
3. Vitičaste zagrade uvek pisati u novom redu.
4. Uvek koristiti vitičaste zagrade kod uslovnih iskaza:
```
if(condition)
{
    ...
```

```
}
```

5. Poravnavati kod koji pripada istom bloku tabulatorom širine četiri prazna mesta.
6. Rekurzivno uvući blokove koda koji se nalaze unutar istog para zagrada.

```
if(condition)
{
    int iterationCount = 0;
    for(int i = 0; i < list.Count(); i++)
    {
        // some processing
    }
}
```

7. Svaku promenljivu deklarisati u posebnom redu.
8. Lokalne promenljive inicijalizovati pri deklaraciji.
9. Referencirati domene imena na vrhu fajla.
10. Sistemske domene imena staviti iznad izvedenih.
11. Ne referencivati domene imena koji se ne koriste.
12. Grupisati članove klasa po tipu u sledećem redosledu:
    - Podaci članovi,
    - Konstruktori i destruktor,
    - Ugnježdene enumeracije, strukture i klase,
    - Propertiji,
    - Metode.
13. Redosled deklaracije članova klasa po vidljivosti:
    - public,
    - protected,
    - internal (samo C#),
    - private.
14. U dilemi uvek težiti čitljivom kodu.
15. Nikada ne koristiti brojeve direktno u kodu već konstante. U slučaju C/C++ jezika moguće je koristiti i predprocesorske direktive.

Rukovanje izuzecima

1. Ne koristiti izuzetke za kontrolu toka ili vraćanje vrednosti iz funkcije.
2. Hvatati samo izuzetke koje je moguće obraditi.
3. Nikada ne deklarisati prazan `catch` blok.
4. Uvek hvatati što određeniji tip izuzetka.
5. Filtere izizetaka navoditi od što određenijeg do opštijih.
6. U `finally` bloku izvršiti jedino odključavanje resursa iz odgovarajućeg `try` bloka,

Komentari

1. Moraju biti na engleskom jeziku, biti gramatički ispravni i imati odgovarajuće interpunkcijske znakove.
2. Pisati ih razumljivo i koncizno.

C# specifična pravila

Opšta pravila

1. Glavni domen imena treba da bude ime projekta i može eventualno biti izdeljen po komponentama,
2. Ne definisati tipove u globalnom domenu imena.

Formatiranje

1. Ne oslanjati se na podrazumevani identifikator vidljivosti već ga uvek navesti.
2. Metode koje pripadaju istom interfejsu grupisati zajedno u oblasti sa `region` direktivom.
3. Dodati ime foldera domenu imena za fajlove u pod folderima.
4. Atribute klase navesti jedne ispod drugog.
5. Izbegavati boxing/unboxing.

Izuzeci

1. Prilikom propagiranja izuzetka očuvati call stack pozivom `throw` bez argumenata,

```
try
{
    ...
}
catch (ArgumentException e)
{
    // Execute custom action
    ...
    // Propagate exception
    throw;
}
```

2. Izbegavati ponovno bacanje izuzetaka već koristiti ugnježdene izuzetke. Kada je moguće proširiti izuzetak dodatnim informacijama.

```
try
{
    ...
}
catch (ArgumentException e)
{
    // Execute custom action
    ...
    // Create exception with nested exception
    Exception ex = new Exception(e);
    ex.Message = "This is additional information."
    throw ex;
}
```

Objekti

1. Klase koje implementiraju `IDisposable` interfejs instancirati unutar `using` bloka,
2. Klase koje koristi resurse van .NET okruženja trebaju da implementiraju `IDisposable` interface.

Sledeća tablica sadrži pravila imenovanja po tipu entiteta:

Identifikator	Public	Protected	Internal	Private	Dodatne informacije
Projektni fajl	P	×	×	×	Potrebno je da bude isto kao i ime biblioteke ili izvršnog fajla i glavni domen imena.
Izvorni fajl	P	×	×	×	Potrebno je da se poklapa sa imenom klase koju sadrži.
Ostali fajlovi	P	×	×	×	
Domen imena	P	×	×	×	Potrebno je da se delimično ili potpuno poklapa sa imenom projekta.
Klasa ili struktura	P	P	P	P	
Interfejs	P	P	P	P	Prefiks veliko I.
Generička klasa	P	P	P	P	Koristiti T ili K kao identifikator tipova.
Metode	P	P	P	P	
Properti	P	P	P	P	Ne koristiti Get i Set prefikse.
Polje	P	P	P	_c	Sva polja trebaju biti private.
Konstanta	P	P	P	_c	
Statičko polje	P	P	P	_c	Sva polja trebaju biti private.
Enumeracija	P	P	P	P	
Delegat	P	P	P	P	
Događaj	P	P	P	P	
Lokalna promenljiva	×	×	×	c	
Parametar	×	×	×	c	

Tabela Error! No sequence specified.**: C# pravila formata imena. Paskal notacija – P, kamilja notacija – c, kamilja notacija sa prefiksom '_' - _c, nije primenljivo - × [1]**

Komentari

1. Za komentarisanje koristiti linijske komentare (//, ///).
2. Kod komentarisanja klasa, podataka članova i metoda koristiti automatski generisan šablon komentara sa popunjenim svim izgenerisanim segmentima.
3. Koristiti CDATA tag gde je moguće.

C/C++ specifična pravila

1. Funkcije bez parametara definisati bez parametara umesto sa jednim void parametrom.

Pravila imenovanja po tipu entiteta su prikazana na sledećoj tabeli:

Identifikator	Public	Protected	Private	Dodatne informacije
Projektni fajl	P	×	×	Potrebno je da bude isto kao i ime biblioteke ili izvršnog fajla i glavni domen imena.
Izvorni fajl	P	×	×	Potrebno je da se poklapa sa imenom klase koju sadrži ili funkcionalnosti u slučanu C programskog jezika.
Ostali fajlovi	P	×	×	
Domen imena	P	×	×	Potrebno je da se delimično ili potpuno poklapa sa imenom projekta.
Klasa, struktura ili tip	P	P	P	
Generička klasa	P	P	P	Koristiti T, T1, T2... kao identifikatore tipova.
Metode	P	P	P	
Polje	P	P	c	Sva polja trebaju biti private.
Konstanta	P	P	c	
Statičko polje	P	P	c	Sva polja trebaju biti private.
Enumeracija	P	P	P	
Lokalna promenljiva	×	×	c	
Parametar	×	×	c	
Predprocesorska direktiva	V	V	V	

Tabela Error! No sequence specified.**: C/C++ pravila formata imena. Paskal notacija – P, kamilja notacija – c, kamilja notacija sa prefiksom '_' - _c, velika slova – V, nije primenljivo - ×**

Komentari

1. Komentari koji dokumentuju podatke članove trebaju biti linijski i odmah iznad entiteta koji dokumentuju. Mogu zauzeti više linija.
   ```
   // map containing mapping between client id and client data
   map<int, void*> clientToDataMapping
   ```
2. Komentari funkcija, metoda članica i klasa trebaju biti blokovski komentari samo u header fajlu.
3. Komentar koji dokumentuje klasu treba biti iznad definicije klase u sledećem formatu:
   ```
   /**
    * @class Temperature
    *
    * @brief Contains single measurement of temperature in system.
    *
    * This class enables conversions between different temperature scales.
    *
    */
   ```
4. Komentar koji dokumentuje klasu treba biti iznad definicije klase u sledećem formatu:
   ```
   /**
    * @struct Temperature
    *
    * @brief  Contains single measurement of temperature in system.
    *
    * This structure enables conversions between different temperature scales.
    *
    */
   ```

5. Komentar koji dokumentuje funkcije ili metode klase se treba nalaziti odmah iznad funkcije u sledećem formatu:
   ```
   /**
    * @brief     Executes validation and calls
    *           sendDMSDataReplicationMessageCallback
    *
    * @param     xfrHeader [in]       - update xfr header
    *
    *            xfrHeaderSize [in]   - xfr header size
    *
    *            update [in]          - replication update
    *
    *            updateSize [in]      - replication update size
    *
    * @return    return value of sendDMSDataReplicationMessageCallback if
    *            called, otherwise false.
    *
    */
   ```
6. Na početku svakog header fajla potrebno je da se nalazi zaglavlje koje navodi koju funkcionalnosti pokrivaju definicije u fajlu. Zaglavlje treba biti sledećeg formata:
   ```
   /**
    *  @file  Processing.h
    *
    *  @brief Functions for processing of process variables.
    *
    */
   ```

Literatura

1. C# Coding Standards for .NET
   Lance Hunt
   Mart 2007