JavaTürk

Java Kod İsimlendirme ve Şekil Standardı

Sürüm 0.2

Akin Kaldiroglu

akin@javaturk.org

www.javaturk.org

İçindekiler

www.javaturk.org

Değişiklik Geçmişi

Tarih	Açıklama	Yapan
24 Mart 2014	İlk oluşturma.	A. Kaldiroglu
23 Temmuz 2014	Bazı ekler ve değişiklikler.	A. Kaldiroglu

JavaTürk Java Kod İsimlendirme ve Şekil Standardı

Giriş

Bu dokümanda, Java kodunda kullanılacak isimlendirme ve şekil (format) standartları sıralanmıştır.

Bu doküman serbestçe kullanılabilir ve en güncel haline www.javaturk.org adresinden erişilebilir.

1. Genel kurallar

1. Daima isimlendirme ve şekil standartlarına uy. Ne kendinin ne de bir başkasının isimleri ve formatı anlamak için enerji harcamasına sebep olma.
2. Daima (standarda uyarak) umulan şekilde kodunu yaz, şasırtma. Standardın dışına çıkman gerektiğinde bunu açıkla.
3. Programındaki her şey önce anlaşılır sonra küçük-kısa olsun. Ama küçüklük-kısalık için anlaşılırlığı feda etme.
4. Hiç bir projeye bu stadartları kullanmadan baslama, başlatma. Yanlış yapılan şeylerin ileride düzeltilmesi çok zordur.

2. En temel şekil kuralları

1. Daima paragraf kullan. Kod yazarken parmakların sıklıkla format tuşlarında olsun. Ne sen ne de bir başkasının formatsız koda bakmasına izin verme.
2. Her satırda sadece bir cümle (statement) yaz.

www.javaturk.org

3. Uzun satırları bir kaç satıra yay ki yatay scrolla ihtiyaç kalmasın.
4. Mekanı rahat kullan: Mümkün olan her yerde boşluk, " ", ve boş satır kullanarak okunurluğu arttır.
5. Ifadelerin öğeleri arasında muhakkak boşluk bırak.

   Böyle yapma:

```
rs = a+b*((c/a)*b);
```

   Böyle yap:

```
rs = a + b * ((c / a)* b);
```

6. Blokları mümkünse "{ }" ile değilse boş bir satır ile ayır. Blok kullanmak için sadece if-while-for gibi yapıları bekleme.
7. Zeka yarışına girme, "=" dahil en az 3 operatörlü ifadeleri anlamak operatörlerin öncelik ve ilişkilendirme bilgisine bağlı olmasın, parantez kullan.

   Böyle yapma:

```
float rs = a + ++b * c/a * b;

a += b += c;
```

   Böyle de yapma:

```
rs = a + (++b)* ((c / a)* b);
```

   Böyle yap:

```
b++;

rs = a + b * ((c / a)* b);


b += c;

a += b;
```

8. Zincirleme üye erişimi ile birleşik ifade yazma, her ifadede bir üyeye eriş.

   Böyle yapma:

```
customer.getCompany().getAddress().getStreet();
```


www.javaturk.org

Böyle yap:

```
Company company = customer.getCompany();

Address address = company.getAddress();

Street street = address. getStreet();
```

Bu şekildeki zincirleme erişimin rahatlıkla kullanılabilecği tek yer web katmanındaki View yapıları (JSP, JSF vb.) olabilir. BU yapılarda EL yardımıyla zincirleme erişim kolaylık sağlamaktadır.

3. Genel isimlendirme kuralları

9. Daima İngilizce isimler kullan ve kesinlikle yanlış yazma, emin değilsen sözlüğe bak.
10. Daima anlamlı isimler kullan; uzun olsun, anlamsız olmasın.

```
addedValueTaxRate, getDefaultAccountInterestRate()
```

11. Okumayı zorlaştıran (özellikle sesli harfleri atarak elde edilen) kısaltmalardan kaçın.

    `qry` yerine `query`, `cstmr` yerine `customer` kullan.
12. Tanıdık isimler kullan. Aynı şeyler için her yerde aynı ismi kullan.
13. İsim verirken herhangi bir kodlama kullanma. Örneğin Hungarian notasyonunu uygulayarak aşağıdaki gibi isimler verme:

```
m_name, d_interest, l_increase
```

    Bu durumun tek istisnası arayüz isimlerindeki “I” olabilir.
14. Kısaltmalarda büyük harflerle yazma.

    Böyle yapma:

```
hTTPSession, TCPIPConnection, getXMLNode(),
getHTTPMethod()
```

    Böyle yap:

www.javaturk.org

```
httpSession, TcpIpConnection, getXmlNode(),
getHttpMethod()
```

15. Tutarlı ol. Aynı ismi sadece küçük-büyük harf ayrımıyla ya da hem kısa hem uzun şekliyle defalarca kullanma.

`sqlQuery, sqlQry` ya da `session, ssn`

16. İsimlendirmede daima "Camel Case" yaklaşımını kullan, alt çizgiden "_", uzak dur. Camel Case'in iki türü vardır, her ismin baş harfinin büyük olduğu Upper Came Case (UCC) ile sadece ilk kelimenin ilk harfinin küçük, sonrasının UCC olarak devam ettiği Lower Camel Case (LCC).

```
StudentInformation is UCC
getAllStudents() is LCC
studentAddress is LCC
```

4. Paket isimlendirme kuralları

17. Paket isimlerine internet alan adınızı tersinden yazarak başla.

`tr.com.selsoft, org.javaturk`

18. Paketlerini küçük harfle yaz ve tek ve tekil isimler ver.

`tr.com.selsoft.atm.domains`

`org.javaturk.designpattern.customers`

19. Paket isimlerin, paket içeriği hakkında bilgi versin.

`tr.com.selsoft.atm.domain, org.javaturk.atm.view.bean`

5. Tip isimlendirme kuralları

20. Sınıf, arayüz, enumeration gibi tiplerin adlandırırken isim kullan ve UCC yaz.

www.javaturk.org

```
Account, CheckingAccountService, StudentInformation
```

21. Arayüzleri adlandırırken daima isim ya da sıfat kullan ve UCC yaz.

```
Payable, ActionListener
```

22. Bir konuyla ilgili özellikleri, sabiteleri ya da metotları bir araya getiren tiplere çoğul isim ver.

```
AtmProperties, StringUtils, ATMProperties
```

23. Enum tiplere tekil isimler ver.

```
Day, Month, Size
```

6. Değişken isimlendirme kuralları

24. Değişken adlandırmalarında isim kullan ve daima LCC yaz.

```
count, firstName, taxRate, orderNumber
```

25. Torbalar için çoğul isimler kullan.

```
Collection<Student> students

Map<Integer, Player> players
```

26. Boolean değişkenler için uygunsa edilgen fiil (ya da sıfat-fiil) kullan öyle ki başına "is" getirildiğinde anlamlı bir soru olsun. Boolean değişken isimlerinde "is" ya da "are" kullanma.

```
married, tankFilled, seatBooked, tasksFinished
```

Eğer boolean değişken sahip olma durumunu gösteriyorsa ismin sonuna "Installed" gibi bir son ek getirilebilir.

```
gasTankSensorInstalled, radioInstalled
```

www.javaturk.org

27. Özellikler (properties) için daima JavaBean (bean) gösterimini kullan. Bean gösteriminde tüm değişkenler "private" (kalıtım durumunda "protected") tanımlanır ve bunlara LCC olarak yazılmış set/get metotları ile ulaşılır:

```
private String name;
public String getName() {
        return name;
}
public void setName(String name) {
        this.name = name;
}
```

Boolean değişkenler için getter olarak "is" ön ekli metot kullanılır:

```
private boolean deceased = false;
public boolean isDeceased() {
        return deceased;
}
public void setDeceased(boolean deceased) {
        this.deceased = deceased;
}
```

28. Metotlardaki yerel değişkenler için nesne değişkenleriyle aynı isimleri kullan ve nesne değişkenlerinie "`this`" ile ulaş.

29. Kurucu ya da set metotlarına nesne değişkeni ile aynı isimde parametre geç, nesne değişkenine "this" ile ulaş. *Bkz. #27*

30. Kısaltılmış ya da bir-iki harflık isimleri sadece sık kullanılan yerel değişkenler için kullan.

Döngü indexleri için `i, j`

`String s` ya da `String str`

www.javaturk.org

stream için `in` ve `out`, exception için `e` ya da `ex`

31. CamelCase yaklaşımının tek istisnası olarak sabitelerde (public, static ve final) araları alt çizgi "_" ile ayrılmış büyük harfli kelimeler kullan. Başka hiç bir isimde "_" kullanma.

```
public static final double ADDED_VALUE_TAX = 0.18;
public static final String USERNAME = "app";
```

7. Metot isimlendirme kuralları

32. Get/set metotlarını JavaBean gösterimiyle yaz. *Bkz. #27*

33. Metot isimlerinde daima emir kipi kullan ve LCC yaz.

```
calculateTax(), findOwnerOfAccount()
```

www.javaturk.org

Kaynaklar

- Java Code Conventions September 12, 1997 (Oracle Java Code Conventions http://www.oracle.com/technetwork/java/codeconv-138413.html)
- http://www.ambysoft.com/downloads/javaCodingStandards.pdf
- http://www.ambysoft.com/downloads/javaCodingStandardsSummary.pdf
- Google Style of Java http://google-styleguide.googlecode.com/svn/trunk/javaguide.html
- Al Vermeulen et al., The Elements of Java Style, CU Press, 2007
- http://collaboratory.emsl.pnl.gov/docs/collab/sam/CodeStandards.html
- http://www.ambysoft.com/downloads/javaCodingStandards.pdf
- http://www.ambysoft.com/downloads/javaCodingStandardsSummary.pdf
- http://www.javacodegeeks.com/2012/10/java-coding-conventions-considered-harmful.html
- http://www.iwombat.com/standards/JavaStyleGuide.html
- Unmaintainable Code http://mindprod.com/jgloss/unmain.html

www.javaturk.org