

Uvod u softversko inženjerstvo

Čist dizajn koda I

Nikola Luburić
nikola.luburic@uns.ac.rs

```
        message =  
        if not hasattr(self, '_headers_buffer'):  
            self._headers_buffer = []  
        self._headers_buffer.append((" %s %d %s\r\n" %  
                                     (self.protocol_version, code, message)).en  
                                     'latin-1', 'strict'))  
  
    def end_header(self, keyword, value):  
        """Send a MIME header to the headers buffer."""  
        if self.request_version != 'HTTP/0.9':  
            if not hasattr(self, '_headers_buffer'):  
                self._headers_buffer = []  
            self._headers_buffer.append(  
                ("%s: %s\r\n" % (keyword, value)).encode('lati  
        keyword.lower() == 'connection':  
            if value.lower() == 'close':  
                self.close_connection = True  
            elif value.lower() == 'keep-alive':  
                self.close_connection = False
```

Diskusija „Značajnih naziva“

Diskusija „Čistih funkcija“

Komentari

Čist dizajn koda I

Refaktorisanje pozorišta

Značajni nazivi

Nazive pronalazimo u svim segmentima razvoja softvera - kod identifikatora promenljive, funkcije, klase, ali i biblioteke i aplikacije. Jasan naziv funkcije nas oslobađa od čitanja njenog tela, dok će misteriozni naziv zahtevati dodatan mentalni napor da razumemo svrhu koncepta koji opisuje. U najgorem slučaju, loš naziv će nas naterati da formiramo pogrešnu pretpostavku, što će nam drastično produžiti vreme razvoja. Kroz ovaj modul ispitujemo dobre i loše prakse za formiranja naziva identifikatora u kodu.

-
- ✓ Konstruiši značajne nazive za identifikatore u kodu
 - ✓ Prati timske konvencije prilikom formiranja naziva
 - ✓ Izbegavaj beznačajne reči
 - ✓ Primeni prikladne tipove reči
 - ✓ Koristi terminologiju domena problema
 - ✓ Analiziraj širi kontekst prilikom formiranja naziva
 - ✓ Formiraj naziv na dobrom nivou apstrakcije
-

Čiste funkcije

Nazive pronalazimo u svim segmentima razvoja softvera - kod identifikatora promenljive, funkcije, klase, ali i biblioteke i aplikacije. Jasan naziv funkcije nas oslobađa od čitanja njenog tela, dok će misteriozni naziv zahtevati dodatan mentalni napor da razumemo svrhu koncepta koji opisuje. U najgorem slučaju, loš naziv će nas naterati da formiramo pogrešnu pretpostavku, što će nam drastično produžiti vreme razvoja. Kroz ovaj modul ispitujemo dobre i loše prakse za formiranja naziva identifikatora u kodu.

-
- ✓ Konstruiši čiste funkcije
 - ✓ Segmentiraj dugačku funkciju spram regiona povezane logike
 - ✓ Umanji složenost funkcije
 - ✓ Reorganizuj logiku funkcije radi smanjenja njene složenosti
 - ✓ Izdvoj složenu logiku u posebnu funkciju
 - ✓ Imenuj rezultat složene logike
 - ✓ Smanji broj parametra funkcije upotrebom prikladne strategije
 - ✓ Odredi semantičku svrhu funkcije
-



Robert Martin (Uncle Bob)

Author of Clean Code

The proper use of comments is to compensate for our failure to express ourselves in code. Comments are always failures. We must have them because we cannot always figure out how to express ourselves without them.

Komentari ne nadoknađuju za loš kod

```
// Check to see if the employee  
// is eligible for full benefits  
if ((employee.flags & HOURLY_FLAG) &&  
    (employee.age > 65))
```

Izrazi komentar kroz kod

```
if (employee.isEligibleForFullBenefits())
```

Kada su komentari korisni?

```
8 // Dear programmer:
9 // When I wrote this code, only god and
10 // I knew how it worked.
11 // Now, only god knows it!
12 //
13 // Therefore, if you are trying to optimize
14 // this routine and it fails (most surely),
15 // please increase this counter as a
16 // warning for the next person:
17 //
18 // total_hours_wasted_here = 254
19 //
20
```

Kada su komentari korisni?

- ❖ Prvi korak u refaktorisanju
- ❖ Objašnjavanje namere

```
//Our best attempt to get a race condition
//by creating a large number of threads.
for (int i = 0; i < 25000; i++) {
    BuilderThread builderThread =
        new BuilderThread(builder, failFlag);
    Thread thread = new Thread(builderThread);
    thread.start();
}
assertEquals(false, failFlag.get());
```


Kada su komentari korisni?

- ❖ Prvi korak u refaktorisanju
- ❖ Objašnjavanje namere
- ❖ Isticanje „osetljivog“ koda

```
public static SimpleDateFormat makeDate () {  
    //SimpleDateFormat is not thread safe, so we  
    //need to create each instance independently.  
    SimpleDateFormat df = new SimpleDateFormat(  
        "EEE, dd MMM  yyyy HH:mm:ss z");  
    df.setTimeZone(TimeZone.getTimeZone("GMT"));  
    return df;  
}
```

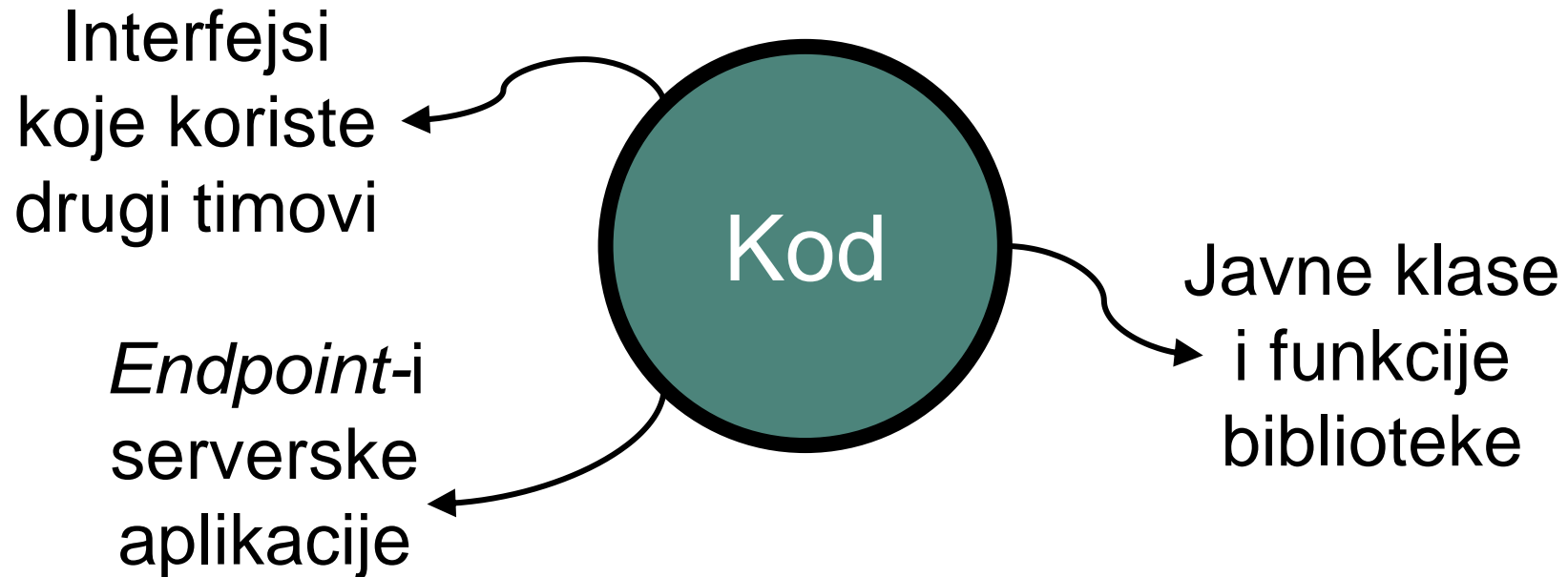
Kada su komentari korisni?

- ❖ Prvi korak u refaktorisanju
- ❖ Objašnjavanje namere
- ❖ Isticanje „osetljivog“ koda

```
// need to reset foo before calling bar due  
// to a bug in the foo component.  
foo.reset()  
foo.bar();
```

Kada su komentari korisni?

- ❖ Prvi korak u refaktorisanju
- ❖ Objašnjavanje namere
- ❖ Isticanje „osetljivog“ koda
- ❖ Dokumentovanje „javnog“ API-a



Šta je loše kod komentara?

❖ Mumlanje

```
public void loadProperties() {  
    try {  
        String propPath = location + PROP_FILE;  
        FileInputStream propertiesStream =  
            new FileInputStream(propPath);  
        loadedProperties.load(propertiesStream);  
    }  
    catch(IOException e) {  
        // No prop file means defaults are loaded  
    }  
}
```

*koja pitanja
ovo povlači?*

Šta je loše kod komentara?

❖ Mumlanje


*previše
informacije*

```
/* RFC 2045 - Multipurpose Internet Mail Extensions (MIME)  
Part One: Format of Internet Message Bodies section 6.8.  
Base64 Content-Transfer-Encoding The encoding process  
represents 24-bit groups of input bits as output strings of 4  
encoded characters. Proceeding from left to right, a 24-bit  
input group is formed by concatenating 3 8-bit input groups.  
These 24 bits are then treated as 4 concatenated 6-bit  
groups, each of which is translated into a single digit in  
the base64 alphabet. When encoding a bit stream via the  
base64 encoding, the bit stream must be presumed to be  
ordered with the most-significant-bit first. The first bit in  
the stream will be the high-order bit in the first 8-bit  
byte, and the eighth bit will be the low-order bit in the  
first 8-bit byte, and so on.*/
```

Šta je loše kod komentara?

- ❖ Mumlanje
- ❖ Redundantnost

```
/**  
 * Port on which fit would run.  
 * Defaults to <b>8082</b>.  
 * @param fitPort  
 */  
public void setFitPort(int fitPort)  
{  
    this.fitPort = fitPort;  
}
```



*nije lokalan
podataka*

Šta je loše kod komentara?

- ❖ Mumlanje
- ❖ Redundantnost

```
/**  
 * @param title The title of the CD  
 * @param author The author of the CD  
 */  
void addCD(String title, String author) {  
    CD cd = new CD(title, author);  
    cdList.add(cd);  
}
```

Šta je loše kod komentara?

- ❖ Mumlanje
- ❖ Redundantnost

```
/**
 * Returns the day of the month.
 *
 * @return the day of the month.
 */
public int getDayOfMonth() {
    return dayOfMonth;
}
```


Šta je loše kod komentara?

- ❖ Mumlanje
- ❖ Redundantnost
- ❖ Laž

```
// Utility method that returns when this.closed is
// true. Throws exception if the timeout reached
synchronized void waitForClose(long timeoutMillis)
    throws Exception {
    if(!closed) {
        wait(timeoutMillis);
        if(!closed)
            throw new Exception("Could not be closed");
    } }
```

Šta je loše kod komentara?

- ❖ Mumlanje
- ❖ Redundantnost
- ❖ Laž

```
MockRequest request;
```

```
private String HTTP_DATE_REGEXP =
```

```
    "[SMTWF][a-z]{2}\\,\\s[0-9]{2}\\s[JFMASOND][a-z]{2}\\s" +
```

```
    "[0-9]{4}\\s[0-9]{2}\\:[0-9]{2}\\:[0-9]{2}\\sGMT";
```

```
private Response response;
```

```
private FileResponder responder;
```

```
private Locale saveLocale;
```

```
// Example: "Tue, 02 Apr 2003 22:18:49 GMT"
```

Šta je zlo kod komentara?

- ❖ Mumlanje
- ❖ Redundantnost
- ❖ Laž
- ❖ Zakomentarisan kod

```
InputStreamResponse response = new InputStreamResponse();  
response.setBody(formatter.getStream(), formatter.getCount());  
// InputStream resultsStream = formatter.getResultStream();  
// StreamReader reader = new StreamReader(resultsStream);  
// response.setContent(reader.read(formatter.getByteCount()));
```

*da li smem ovo
da obrišem?*