3

Funkcije

U ranim danima programiranja sklopili smo naše prve sisteme rutina i podrutina. Zatim, u doba Fortrana i PL/1, sastavili smo naše sisteme programa, potprogram i funkcija. Od tih početaka, preživele su samo funkcije, kao prva linija organizacije u bilo kom programu. Tema ovog poglavlja je pisanje funkcija na dobar način.

Razmotrite kod u listingu 3-1. Teško je pronaći dugačku funkciju u alatu FitNesse[1], ali nakon malo pretraživanja naći ćete je. Ne samo da je dugačka, već ima duplirani kod, puno čudnih nizova, mnogo neobičnih i neočiglednih tipova podataka i interfejsa za programiranje aplikacija (eng. *application programming interface, API*). U naredna tri minuta pogledajte koliko smisla možete da nađete u svemu tome.

Listing 3-1
HtmlUtil.java (FitNesse 20070619)

```
public static String testableHtml(
  PageData pageData,
boolean includeSuiteSetup
) throws Exception {
  WikiPage wikiPage = pageData.getWikiPage();
  StringBuffer buffer = new StringBuffer();
  if (pageData.hasAttribute("Test")) {
    if (includeSuiteSetup) {
      WikiPage suiteSetup =
        PageCrawlerImpl.getInheritedPage(
                SuiteResponder.SUITE _ SETUP _ NAME, wikiPage
        );
      if (suiteSetup != null) {
        WikiPagePath pagePath =
          suiteSetup.getPageCrawler().getFullPath(suiteSetup);
        String pagePathName = PathParser.render(pagePath);
        buffer.append("!include -setup .")
              .append(pagePathName)
              .append("\n");
      }
    }
    WikiPage setup =
      PageCrawlerImpl.getInheritedPage("SetUp", wikiPage);
    if (setup != null) {
      WikiPagePath setupPath =
        wikiPage.getPageCrawler().getFullPath(setup);
      String setupPathName = PathParser.render(setupPath);
      buffer.append("!include -setup .")
            .append(setupPathName)
            .append("\n");
    }
  }
  buffer.append(pageData.getContent());
  if (pageData.hasAttribute("Test")) {
    WikiPage teardown =
      PageCrawlerImpl.getInheritedPage("TearDown", wikiPage);
    if (teardown != null) {
```

1. Alat za testiranje otvorenog koda, www.fitnesse.org

Listing 3-1 (nastavak)
HtmlUtil.java (FitNesse 20070619)

```
      WikiPagePath tearDownPath =
        wikiPage.getPageCrawler().getFullPath(teardown);
      String tearDownPathName = PathParser.render(tearDownPath);
      buffer.append("\n")
            .append("!include -teardown .")
            .append(tearDownPathName)
            .append("\n");
    }
    if (includeSuiteSetup) {
      WikiPage suiteTeardown =
        PageCrawlerImpl.getInheritedPage(
                SuiteResponder.SUITE _ TEARDOWN _ NAME,
                wikiPage
        );
      if (suiteTeardown != null) {
        WikiPagePath pagePath =
          suiteTeardown.getPageCrawler().getFullPath (suiteTeardown);
        String pagePathName = PathParser.render(pagePath);
        buffer.append("!include -teardown .")
              .append(pagePathName)
              .append("\n");
      }
    }
  }
  pageData.setContent(buffer.toString());
  return pageData.getHtml();
}
```

Da li posle tri minuta proučavanja razumete ovu funkciju? Verovatno ne. Previše toga se ovde događa na previše različitih nivoa apstrakcije. Postoje neobični nizovi i čudni pozivi funkcije pomešani sa dvostruko ugnežđenim `if` naredbama kojima upravljaju logičke vrednosti.

Međutim, sa samo nekoliko jednostavnih metoda ekstrakcije, nekim preimenovanjima i malim restrukturiranjem, uspeo sam da uhvatim nameru funkcije u devet linija u listingu 3-2. Pogledajte u naredna tri minuta da li možete to da shvatite.

Verovatno ne razumete sve detalje, osim ako niste koristili FitNesse. Ipak, verovatno razumete da ova funkcija uključuje početne vrednosti (eng. *setup*) i raščlanjivanje (eng. *teardown*) test stranice i zatim pravi HTML stranu. Ako ste upoznati sa JUnitom[2] verovatno shvatate da ova funkcija pripada nekoj vrsti radnog okruženja za testiranje veb strana. To je tačno. Dolaženje do tih informacija na osnovu listinga 3-2 prilično je jednostavno, ali je prilično nejasno u listingu 3-1.

2. Alat za testiranje otvorenog koda u Javi. www.junit.org

Šta je to što funkciju iz listinga 3-2 čini čitkom i razumljivom? Kako da učinimo da funkcija saopšti svoju nameru? Koje osobine možemo dati funkcijama tako da povremenom čitaocu omoguće da oseti vrstu programa koje one sadrže?

Listing 3-2
HtmlUtil.java (refaktorisano)

```
public static String renderPageWithSetupsAndTeardowns(
  PageData pageData, boolean isSuite
) throws Exception {
  boolean isTestPage = pageData.hasAttribute("Test");
  if (isTestPage) {
    WikiPage testPage = pageData.getWikiPage();
    StringBuffer newPageContent = new StringBuffer();
    includeSetupPages(testPage, newPageContent, isSuite);
    newPageContent.append(pageData.getContent());
    includeTeardownPages(testPage, newPageContent, isSuite);
    pageData.setContent(newPageContent.toString());
  }

  return pageData.getHtml();
}
```

Mala!

Prvo pravilo funkcija je da treba da budu male. Drugo pravilo funkcija je *da bi trebalo da budu manje i od toga*. Ovo nije tvrdnja koju mogu da opravdam. Ne mogu da dam ni jednu referencu koja pokazuje da su vrlo male funkcije bolje, ali ono što mogu da kažem je da već skoro četiri decenije pišem funkcije različitih veličina. Napisao sam nekoliko gadosti od 3.000 linija koda. Napisao sam mnogo funkcija u opsegu od 100 do 300 linija koda, ali i funkcije koje su bile dužine od 20 do 30 linija koda. Dugi niz pokušaja i grešaka naučio me je da bi funkcije trebalo da budu veoma male!

Osamdesetih godina prošlog veka govorili smo da funkcija ne treba da bude veća od veličine ekrana. Naravno da je to rečeno u vreme kada su ekrani VT100 bili veličine 24 reda i 80 kolona, a editori su za svoje potrebe koristili 4 reda. Danas sa manjim fontovima i lepim velikim monitorom, na ekran može da stane 150 znakova u jednom redu i 100 ili više redova. Ipak, redovi ne smeju da budu dugački 150 znakova, a funkcije ne bi trebalo da budu dugačke 100 redova. Ustvari, funkcije nikako ne bi trebalo da budu duže od 20 redova.

Koliko kratka bi trebalo da bude funkcija? 1999. godine posetio sam Kenta Becka u njegovoj kući u Oregonu. Zajedno smo napisali nekoliko programa, a on mi je u jednom trenutku pokazao simpatičan mali program napisan u jeziku Java/Swing koji je nazvao Sparkle. Program je na ekranu proizveo vizuelni efekat vrlo sličan onome koji stvara čarobnom štapić dobre vile u filmu Pepeljuga. Pomeranjem miša, iz kursora bi iskakale sjajne iskre i padale na dno ekrana u simuliranom gravitacionom polju. Kad mi je Kent pokazao kod, zadivilo me je koliko su sve funkcije bile male. Navikao sam na

funkcije napisane u programu Swing koje su bile dugačke „kilometrima“. Svaka funkcija u ovom programu bila je dugačka samo dva, tri ili četiri reda. Svaka od njih je bila očigledna i jasna sama po sebi. I svaka od njih vas je vodila do sledeće, ubedljivim redosledom. Tako kratke bi trebalo da budu vaše funkcije![3]

Koliko kratke treba da budu vaše funkcije? Obično bi trebalo da budu kraće od listinga 3-2! Zaista, listing 3-2 bi trebalo skratiti na listing 3-3.

Listing 3-3
HtmlUtil.java (re-refactored)

```
public static String renderPageWithSetupsAndTeardowns(
PageData pageData, boolean isSuite) throws Exception {
if (isTestPage(pageData))
includeSetupAndTeardownPages(pageData, isSuite);
return pageData.getHtml();
}
```

Blokovi i uvlačenja

Podrazumeva se da bi blokovi unutar naredbi `if`, `else`, `while` itd. trebalo da budu dugački jedan red. Taj red bi verovatno trebalo da predstavlja pozivanje funkcije. To ne samo da čini funkciju malom, već joj dodaje i dokumentacionu vrednost jer funkcija koja se poziva unutar bloka može imati lepo opisno ime.

Ovo podrazumeva da funkcije ne bi trebalo da budu toliko velike da bi sadržale ugnežđene strukture. Prema tome, unutar funkcije ne sme biti više od jednog ili dva nivoa uvlačenja, što, naravno, čini funkcije lakšim za čitanje i razumevanje.

Činite jednu stvar

Trebalo bi da bude jasno da listing 3-1 radi puno više od jedne stvari. Između ostalog, on pravi bafere, preuzima stranice, traži nasleđene stranice, prikazuje putanje, dodaje prikrivene nizove i generiše HTML kod. Listing 3-1 je veoma zauzet radeći razne stvari, dok sa druge strane, listing 3-3 radi jednu jednostavnu stvar – uključuje početne vrednosti i raščlanjivanja u test stranice.

Sledeći saveti se pojavljuju u jednom ili drugom obliku već 30 ili više godina.

Funkcije treba da rade jednu stvar. Treba da je rade dobro i treba da rade samo to.

3. Pitao sam Kenta da li još uvek ima taj kod, ali nije uspeo da ga pronađe. Pregledao sam i sve svoje stare računare, ali bezuspešno. Ostalo mi je samo sećanje na taj program.