

Dokonalejší chování

Použití knihovných tříd pro implementaci
dokonalejší funkcionality

Překlad originální prezentace ke Kapitole 5

„More-sophisticated behavior“ z učebnice

Objects First with Java - A Practical Introduction using
BlueJ, © David J. Barnes, Michael Kölling

Hlavní pojmy

- Použití knihovných tříd
- Čtení dokumentace
- Psaní dokumentace
- Třídy: `String`, `ArrayList`, `Random`,
`HashMap`, `HashSet`, `Iterator`,
`Arrays`

Knihovna tříd v jazyce Java

- Tisíce tříd.
- Desítky tisíc metod.
- Mnoho užitečných tříd, které programátorovi usnadňují život.
- Knihovnní třídy jsou často provázány.
- Knihovnní třídy jsou organizovány do balíčků.

Práce s knihovnou

- Kompetentní programátor v jazyce Java musí být schopen pracovat s knihovnami.
- Měli byste:
 - znát jména některých důležitých tříd;
 - znát, jak se dozvědět o dalších třídách a jak vyhledat detailní informace.
- Pamatujte si:
 - potřebujeme znát pouze *interfejs* (veřejné rozhraní třídy) a nikoliv implementaci (zdrojový kód třídy).

System technické podpory

- Textový, interaktivní dialogový systém.
- Vychází z programu '*Eliza*' Josepha Weizenbauma (MIT, 1960s)
- <http://www.cse.buffalo.edu/~rapaport/572/S02/weizenbaum.eliza.1966.pdf>
- Prozkoumejte projekt *tech-support-complete...*

Projekt tech-support-complete

- DEMO
- run:
- Welcome to the DodgySoft Technical Support System.
- Please tell us about your problem.
- We will assist you with any problem you might have.
- Please type 'bye' to exit our system.
- > I have problem
- No other customer has ever complained about this before.
- What is your system configuration?
- > I have Windows 10
- This is a known bug to do with the Windows operating system. Please
- report it to Microsoft. There is nothing we can do about this.
- > OK
- That is explained in the manual. Have you read the manual?
-

Struktura hlavního cyklu

```
boolean ukonceno= false;
```

```
while (!ukonceno) {
```

```
    něco proved'
```

```
    if (podmínka ukončení) {
```

```
        ukonceno = true;
```

```
    }
```

```
    else {
```

```
        udělej něco dalšího
```

```
    }
```

```
}
```

Obecný
implementační
vzor pro
opakování.

Tělo hlavního cyklu

```
String input = reader.getInput();  
...  
String response =  
    responder.generateResponse(input);  
System.out.println(response);
```


Podmínka ukončení cyklu

```
String input = reader.getInput();
```

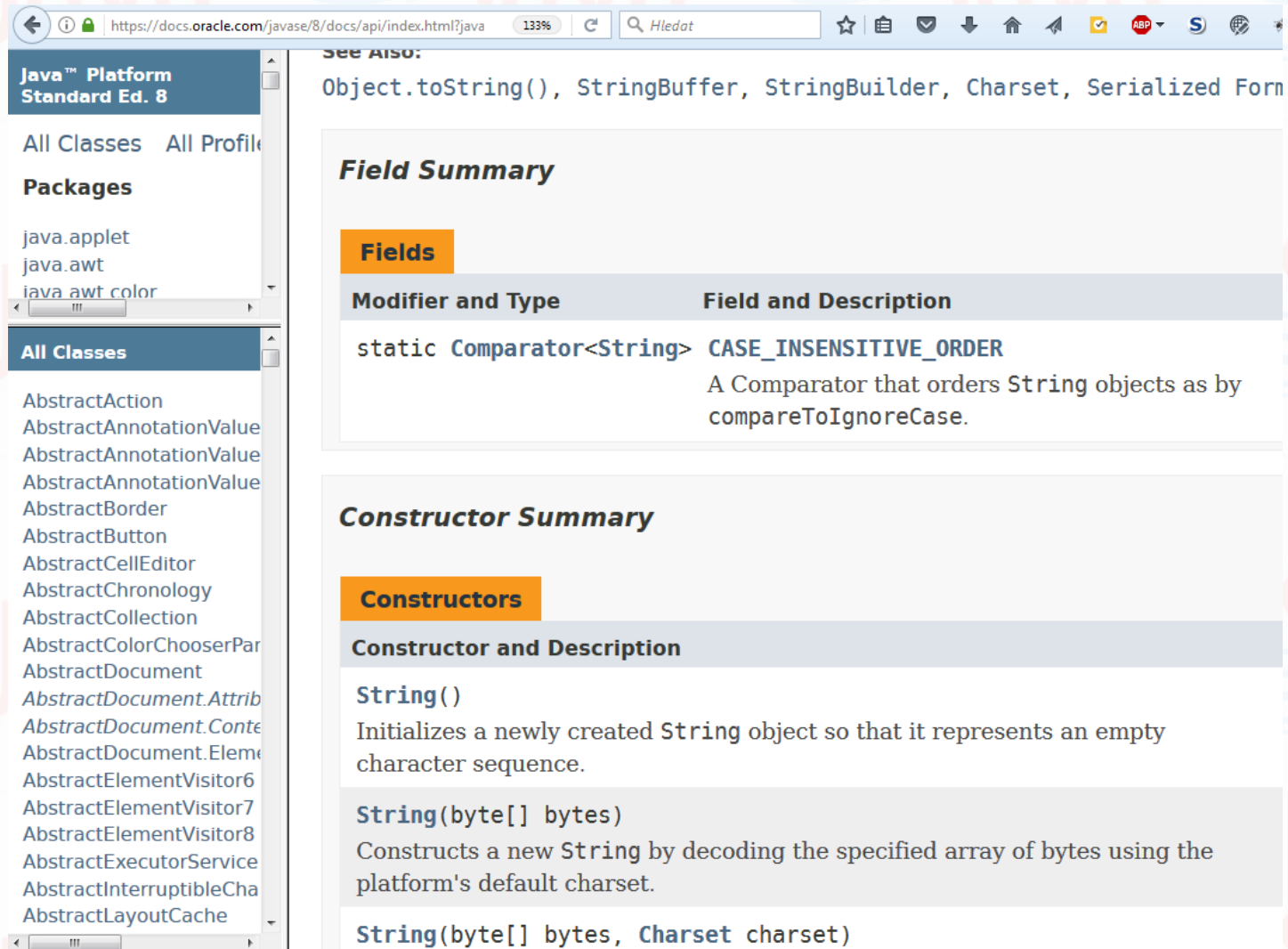
```
if (input.startsWith("bye")) {  
    finished = true;  
}
```

- Odkud pochází metoda **startsWith**?
- Co je to? Co to dělá?
- Jak můžeme vyhledat více informací
 - o **startsWith** ?

Čtení dokumentace třídy

- Dokumentace knihoven je v HTML formátu.
- Čitelná ve webovském prohlížeči.
- API třídy: *Application Programmers' Interface*.
- Popis interfejsu (veřejné rozhraní třídy) pro všechny knihovnické třídy.
- <https://docs.oracle.com/javase/8/docs/api/>

Dokumentace knihovních tříd



See also: `Object.toString()`, `StringBuffer`, `StringBuilder`, `Charset`, `Serialized Form`

Field Summary

Fields	
Modifier and Type	Field and Description
static <code>Comparator<String></code>	<code>CASE_INSENSITIVE_ORDER</code> A <code>Comparator</code> that orders <code>String</code> objects as by <code>compareToIgnoreCase</code> .

Constructor Summary

Constructors
Constructor and Description
<code>String()</code> Initializes a newly created <code>String</code> object so that it represents an empty character sequence.
<code>String(byte[] bytes)</code> Constructs a new <code>String</code> by decoding the specified array of bytes using the platform's default charset.
<code>String(byte[] bytes, <code>Charset</code> charset)</code>

Interfejs versus implementace

Dokumentace obsahuje:

- jméno třídy,
- celkový popis třídy (dokumentační komentář třídy),
- seznam veřejných proměnných (instančních a statických),
- seznam konstruktorů a metod,
- návratové hodnoty a parametry pro konstruktory a metody,
- popis účelu každé metody a konstruktoru.

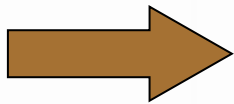


interfejs třídy

Interfejs versus implementace

Dokumentace nezahrnuje:

- privátní instanční proměnné (většina je privátní)
- privátní metody
- těla (zdrojový kód) metod



implementace třídy

Dokumentace pro metodu startsWith

- **startsWith**
 - `public boolean startsWith(String prefix)`
- Tests if this string starts with the specified prefix.
- Parameters:
 - **prefix** - the prefix.
- Returns:
 - **true** if the character sequence represented by the argument is a prefix of the character sequence represented by this string; **false** otherwise. Note also that true will be returned if the argument is an empty string or is equal to this String object as determined by the [equals\(Object\)](#) method.

Metody ze třídy String

- `contains`
- `endsWith`
- `indexOf`
- `substring`
- `toUpperCase`
- `trim`
- Pozor na to, že instance třídy **String** jsou nemodifikovatelné (*immutable*) objekty!!

Použití knihovních tříd

- Třídy jsou organizovány do balíčků.
- Knihovnní třídy musí být *importovány* s využitím příkazu **import** (kromě tříd z balíčku **java.lang**).
- Potom je možné tyto třídy použít jako třídy z aktuálního projektu.

Balíčky a příkaz import

- Importovat můžeme jednu třídu:

```
import java.util.ArrayList;
```

- Importovat můžeme celý balíček:

```
import java.util.*;
```

- Import nemá za následek vložení zdrojového kódu.

Použití třídy Random

- Knihovná třída **Random** může být použita pro generování pseudonáhodných čísel.

```
import java.util.Random;  
...  
Random rand = new Random();  
...  
int num = rand.nextInt();  
int value = 1 + rand.nextInt(100);  
int index = rand.nextInt(list.size());
```

Výběr pseudonáhodných odpovědí

```
public Responder()
{
    randomGenerator = new Random();
    responses = new ArrayList<String>();
    fillResponses();
}

public void fillResponses()
{
    fill responses with a selection of response strings
}

public String generateResponse()
{
    int index =
        randomGenerator.nextInt(responses.size());
    return responses.get(index);
}
```

Parametrizované (generické) třídy

- Dokumentace obsahuje informaci o *typovém parametru/ech* v hlavičce třídy.
 - příklad: **ArrayList<E>**
- Tato jména parametrů se znovu objevují v hlavičkách metod na místě návratového typu nebo jako typ parametru metody:
 - **E get(int index)**
 - **boolean add(E e)**

Parametrizované (generické) třídy

- Jména typových parametrů v generických třídách představují zástupné symboly, které se při použití nahrazují typovými argumenty.

– **`ArrayList<TicketMachine>`**

- `TicketMachine get(int index)`
- `boolean add(TicketMachine e)`

Přehled

- Java má rozsáhlou knihovnu tříd.
- Dobrý programátor musí být obeznámen s touto knihovnou.
- Dokumentace nám říká, co musíme znát, abychom mohli třídu použít (její interfejs).
- Některé třídy jsou parametrizované pomocí typových parametrů.
 - Parametrizované třídy definují parametrizované (generické) typy.

Dokonalejší chování

Použití knihovních tříd pro implementaci
dokonalejší funkcionality

Hlavní pojmy

- Další knihovní třídy
 - **HashSet<E>**
 - **HashMap<K, V>**
- Psaní dokumentace
 - **javadoc**

Použití množin

```
import java.util.HashSet;
```

```
...
```

```
HashSet<String> mySet = new HashSet<String>();
```

```
mySet.add("one");
```

```
mySet.add("two");
```

```
mySet.add("three");
```

```
for(String element : mySet) {  
    do something with element  
}
```

**Porovnejte s
kódem pro
ArrayList!**

Rozdělování řetězců

```
public HashSet<String> getInput()  
{  
    System.out.print("> ");  
    String inputLine =  
        reader.nextLine().trim().toLowerCase();  
  
    String[] wordArray = inputLine.split(" ");  
    HashSet<String> words = new HashSet<String>();  
  
    for(String word : wordArray) {  
        words.add(word);  
    }  
    return words;  
}
```

- <https://docs.oracle.com/javase/tutorial/essential/regex/index.html>

Mapy(Slovníky)

- Slovníky jsou kolekce, které obsahují uspořádané dvojice hodnot.
- Dvojice se skládají z klíče(**key**) a hodnoty (**value**).
- Vyhledávání funguje tak, že pro daný klíč se získá odpovídající hodnota.
- Příklad: telefonní seznam.

Použití slovníků

- Slovník, v němž jsou klíče i hodnoty typu String

:HashMap

"Charles Nguyen"	"(531) 9392 4587"
"Lisa Jones"	"(402) 4536 4674"
"William H. Smith"	"(998) 5488 0123"

Použití slovníků

```
HashMap <String, String> phoneBook =  
    new HashMap<String, String>();  
  
phoneBook.put("Charles Nguyen", "(531) 9392 4587");  
phoneBook.put("Lisa Jones", "(402) 4536 4674");  
phoneBook.put("William H. Smith", "(998) 5488  
    0123");  
  
String phoneNumber = phoneBook.get("Lisa Jones");  
System.out.println(phoneNumber);
```

List, Map a Set

- Alternativní způsoby grupování objektů.
- Jsou k dispozici různé implementace:
 - **ArrayList<E>**, **LinkedList<E>**
 - **HashSet<E>**, **TreeSet<E>**
- Ale **HashMap<K, V>** nesouvisí s **HashSet<E>** navzdory podobným jménům.
- Druhé slovo ve jménu odhaluje souvislost s kategorií kolekce.

Přehled

- Java má rozsáhlou knihovnu tříd.
- Dobrý programátor se musí seznámit s touto knihovnou.
- Dokumentace nám říká, co potřebujeme znát, abychom mohli třídu použít (interfejs).
- Implementace je skryta (ukrývání informací).
- Velmi využívanými třídami jsou kolekce.