



Začínáme programovat v jazyku **JAVA**

- » Nepředpokládá žádné předchozí znalosti programování
- » Neomezuje se na výuku kódování, ale učí, čtenáře programovat podle moderních zásad a metodik
- » Probírá všechny důležité konstrukce jazyka včetně těch, které začátečnické učebnice přeskakují
- » Všechny probírané konstrukce vysvětluje na příkladech
- » Pro výklad složitějších programových konstrukcí používá syntaktické diagramy



edice
začínáme s ...

Začínáme programovat v jazyku JAVA

**RUDOLF PECINOVSKÝ
JARMILA PAVLÍČKOVÁ**

GRADA Publishing



Upozornění pro čtenáře a uživatele této knihy

Všechna práva vyhrazena. Žádná část této tištěné či elektronické knihy nesmí být reprodukována a šířena v papírové, elektronické či jiné podobě bez předchozího písemného souhlasu nakladatele.

Neoprávněné užití této knihy bude **trestně stíháno**.

Rudolf Pecinovský, Jarmila Pavlíčková

Začínáme programovat v jazyku Java

Vydala Grada Publishing, a.s.
U Průhonu 22, Praha 7
obchod@grada.cz, www.grada.cz
tel.: +420 234 264 401
jako svou 8318. publikaci

Odpovědný redaktor Petr Somogyi
Fotografie na obálce Depositphotos/gdolgikh
Grafická úprava a sazba Rudolf Pecinovský
Počet stran 400
První vydání, Praha 2021
Vytiskla TISKÁRNA V RÁJLI, s.r.o, Pardubice

© Grada Publishing, a.s., 2021
Cover Design © Grada Publishing, a. s., 2021
Cover Photo © Depositphotos/gdolgikh

Názvy produktů, firem apod. použité v knize mohou být ochrannými známkami nebo registrovanými ochrannými známkami příslušných vlastníků.

ISBN 978-80-271-4641-3 (ePub)
ISBN 978-80-271-4640-6 (pdf)
ISBN 978-80-271-3062-7 (print)

Všem, kteří se chtějí něco naučit

Stručný obsah

Úvod	19
Část A Superzáklady	27
1 Předehra	28
2 Prostředí JShell	40
3 Zadávání hodnot	51
4 Proměnné a výrazy	65
Část B Začínáme programovat	79
5 Práce s objekty	80
6 Robot Karel a jeho svět, knihovny, balíčky	90
7 Definice metod	103
8 Opakování – cykly	120
9 Rozhodování	130
Část C Objektově orientované programování	143
10 Základy definice třídy	144
11 Stručný úvod do BlueJ	159
12 Uspořádání kódu	169
13 Programátorská dokumentace	180
14 Rozhraní a interfejs	189
15 Dědění interfejsů	200
16 Návrhové vzory	214
17 Prohlubujeme znalosti	229
18 Lambda-výrazy, funkční interfejsy a generické typy	243
19 Dědění implementace	261
20 Abstraktní třídy	276

Část D Knihovny 293

21 Speciální datové typy.....	294
22 Výjimky.....	312
23 Kontejnery	325
24 Pole.....	339
25 Interní datové typy	352
26 Datovody.....	361
27 Čtení a ukládání dat	372
28 Vytvoření aplikace.....	388

Literatura 394

Rejstřík 396

Část E Přílohy 401

A Příprava spuštění JShell pod Windows.....	402
B Význam cizích slov	407
C Česko-americký slovník.....	409

Část F SEZNAMY 411

Seznam výpisů programů.....	412
Seznam obrázků	416
Seznam tabulek	418
Seznam odboček – podšeděných bloků	419

Podrobný obsah

Úvod	19
Komu je kniha určena	20
Koncepce výkladu	20
Potřebné vybavení	21
Doprovodné programy	22
Použité typografické konvence	22
Odbočka – podšeděný blok	24
Zpětná vazba	24
Část A Superzáklady	27
1 Předehra	28
1.1 Trocha historie	28
Co je to program	29
Postupná změna priorit	29
Vývoj metodik programování	31
OOP a OFP	32
1.2 Překladače, interprety, platformy	32
Operační systém a platforma	33
Programovací jazyky	33
Způsoby zpracování programu	33
Překládaný program	34
Interpretovaný program	34
Porovnání	34
Hybridní programy	34
1.3 Java a její zvláštnosti	35
Java je jazyk i platforma	35
1.4 Vývojové nástroje	36
Základní vývojářská sada	36
IDE	37
JShell	37
BlueJ	37
NetBeans	38
IntelliJ IDEA	38
Eclipse	38
Visual Studio Code	39
Shrnutí	39
2 Prostředí JShell	40
2.1 Prostředí JShell	40
Spuštění programu JShell	40
2.2 Úryvky (snippets)	42
Použití proměnných	43
Identifikace úryvků	43

Terminologie: výrazy, příkazy, deklarace, definice	43
Středník	44
Více objektů na řádku, zavlečené chyby	44
2.3 Příkazy prostředí JShell	45
Vyloučení úryvku: <code>/drop</code>	46
Přehled aktivních úryvků: <code>/list</code>	46
Přehled všech úryvků: <code>/list -all</code>	47
Uložení aktivních úryvků: <code>/save <file></code>	48
Uložení všech zadaných úryvků: <code>/save -all <file></code>	48
Uložení dosavadního průběhu seance: <code>/save -history <file></code>	48
Načtení skriptu: <code>/open <file></code>	48
Ukončení seance: <code>/exit</code>	49
Restart: <code>/reset</code>	49
Znovuzavedení: <code>/reload -restore</code>	49
Nastavení startovního skriptu: <code>/set -start <file></code>	49
Nápověda: <code>/?</code>	50
Spuštění editoru: <code>/edit</code>	50
Nastavení běhového prostředí: <code>/env</code>	50
Další informace	50
3 Zadávání hodnot	51
3.1 Datové typy	51
Primitivní datové typy	52
Objektové datové typy	53
Odkazy na objekty	54
3.2 Komentáře	54
3.3 Zadávání celočíselných hodnot	55
3.4 Zadávání reálných hodnot	56
Číslo v exponentovém tvaru	56
3.5 Zadávání znaků	57
3.6 Prázdný odkaz null	59
3.7 Hodnoty typu String	59
Textové bloky	62
3.8 Zadávání logických hodnot	63
3.9 Literály	64
4 Proměnné a výrazy	65
4.1 Pravidla pro tvorbu identifikátorů	65
Používání znaku <code>\$</code>	66
4.2 Deklarace proměnných	66
4.3 Terminologie	68
4.4 Operace přiřazení =	69
Asociativita	69
4.5 Aritmetické operace	70
Sčítání	70
Odčítání	70
Násobení	70
Dělení	70
4.6 Přetypování	71
4.7 Volání metod	72
4.8 Závorky	73
4.9 Složené přiřazení	73
4.10 Inkrementační a dekrementační operátory	74
4.11 Porovnávací operátory	76
4.12 Příkazy versus výrazy	77

Část B Začínáme programovat	79
5 Práce s objekty	80
5.1 Nejprve trocha teorie	80
Principy OOP	80
Objekty	81
Atributy	81
Zprávy	82
Metody	82
Třídy a jejich instance	83
Interní datové typy	84
Třída jako datový typ	84
Třída jako objekt	84
Kontejner	85
Terminologie objektových datových typů	85
5.2 Práce s atributy	86
5.3 Volání metody	86
5.4 Konstruktory a tovární metody	88
Tovární metoda	88
Typické vytvoření instance	89
Pokračování	89
6 Robot Karel a jeho svět, knihovny, balíčky	90
6.1 Robot Karel a jeho svět	90
Vytváření	90
Historie robota Karla	91
Akce	91
Testy	92
Zrychlování	93
Reakce na zavření okna	94
6.2 Knihovny	96
6.3 Velké programy a jejich problémy	97
6.4 Balíčky	98
Kořenový balíček a strom balíčků	98
6.5 Import objektových typů	98
Příklad	99
Import všech typů z daného balíčku – hvězdičkový import	101
6.6 Zakázaný balíček java	102
7 Definice metod	103
7.1 Zásada DRY	103
7.2 Definice a volání metody	104
Povinný středník	104
Příklad jednoduché metody pro robota	104
7.3 Blok příkazů	106
7.4 Metody s parametry	106
Parametry	106
Argumenty	107
Metody robota versus metody prostředí <i>JShell</i>	108
7.5 Metody s více parametry	108
Tisk na standardní výstup	109
7.6 Lokální proměnné	110
Proměnné lokální v bloku	110
7.7 Přetěžování metod	110
7.8 Ještě jednou robot Karel a jeho svět	111
Třídy <code>RobotWorld</code> a <code>RobotWindow</code>	112
Přetížené tovární metody tříd <code>RobotWorld</code> a <code>RobotWindow</code>	112

RobotWorld newWorld()	112
RobotWorld newWorld(int rows, int cols)	112
RobotWorld newWorld(String... world)	112
Společné vlastnosti	112
static void destroyWorld()	113
void destroy()	113
static RobotWorld getWorld()	113
Přetížené konstruktory třídy Karel	113
Karel(int row, int col, Direction dir)	113
Karel(int row, int col, Direction dir, Color color)	113
Karel()	113
7.9 Metody vracející hodnotu	114
7.10 Přehled aktuálně definovaných metod v JShell	116
7.11 Logické výrazy	116
Příklad	117
8 Opakování – cykly	120
8.1 Cykly	120
8.2 Cyklus s počáteční podmínkou – cyklus while	121
8.3 Cyklus s koncovou podmínkou – cyklus do...while	123
8.4 Cyklus s parametrem – cyklus for	124
Příklad	126
putNMarkers(Karel, int)	126
putIncreasingMarkers(Karel)	127
markers(Karel)	127
Test	128
8.5 Nekonečný cyklus	128
8.6 Cyklus s podmínkou uprostřed	129
9 Rozhodování	130
9.1 Jednoduchý podmíněný příkaz	130
9.2 Úplný podmíněný příkaz	132
9.3 Podmíněný výraz	133
9.4 Složený podmíněný příkaz	133
9.5 Přepínač – příkaz/výraz switch	136
Pravidla	136
Přepínací výraz – výraz switch	139
9.6 Cyklus s podmínkou uprostřed – příkaz break	140

Část C Objektově orientované programování 143

10 Základy definice třídy	144
10.1 Vytváříme vlastní třídu	144
Bílé znaky a uspořádání programu	145
10.2 Viditelnost tříd a jejich členů: public, private	145
10.3 Výměna knihoven – knihovna tvarů	146
IO	146
Canvas	146
NamedColor	147
Direction	147
Ellipse, Rectangle, Triangle	147
10.4 Konstruktory	147
Podrobnosti o konstruktorech	149
10.5 Definice atributů	149

Účel atributů	150
10.6 Modifikátor final	150
10.7 Konstruktory a parametr this	151
Kvalifikace parametrem this	152
10.8 Magické hodnoty	152
10.9 Modifikátor static	153
Definice třídy počítající své instance	154
10.10 Zděděné metody, metoda toString()	155
10.11 Výsledná definice a test	155
Upravená definice	155
Test upravené definice	157
11 Stručný úvod do BlueJ	159
11.1 Instalace BlueJ	159
11.2 Projekty a BlueJ	160
Vyhledání a otevření projektu	160
11.3 Aplikační okno BlueJ	161
Okna balíčků	161
11.4 Diagram tříd	163
11.5 Překlad a jeho pravidla	163
11.6 Manipulace s třídami v diagramu	164
11.7 Otevření okna jiného balíčku	167
11.8 Práce se zdrojovými soubory	167
12 Uspořádání kódu	169
12.1 Uspořádání projektů v Javě	169
Uspořádání projektu 67_Java	169
Nové umístění zdrojů	170
12.2 Samostatně definované třídy	170
12.3 Příkazy package a import	170
12.4 Zapouzdření a skrývání implementace	172
12.5 Entity programu	173
12.6 Rozhraní versus implementace	173
Signatura versus kontrakt	174
12.7 Atributy versus vlastnosti; přístupové metody	174
Možné kombinace a výhody	175
Pravidla pro názvy	175
12.8 Uspořádání jednotlivých prvků v těle třídy	176
Motivace pro zavedení některých zásad	176
Kódování	176
Co dřív: atributy nebo metody?	177
Statické a instanční členy	177
Veřejné, neveřejné, soukromé	177
Uvození oddílů	177
12.9 Prázdná standardní třída	178
13 Programátorská dokumentace	180
13.1 Komentáře a dokumentace	180
Proč psát srozumitelné programy	180
Odsazování	182
Dokumentační komentáře	182
13.2 Pomocné značky pro tvorbu dokumentace	183
13.3 Dokumentace balíčku	184
13.4 Ukázka dokumentované třídy	184
13.5 Zobrazení dokumentace	186
Obsah a uspořádání dokumentace	186

	Dokumentace entit vyšší úrovně	186
	Dokumentace datových typů	187
	Vyhledávání	187
	Zavrženě (nedoporučované) entity	187
	13.6 Zakomentování a odkomentování části programu	188
14 Rozhraní a interfejs		189
14.1	Motivace	189
14.2	Rozhraní versus interfejs versus interface	190
	Interfejs a jeho instance	190
14.3	Použití v programu	191
	Knihovní balíček <code>eu.pedu.b67.canvas_2</code>	192
14.4	Použití interfejsu na příkladu	193
	Počáteční úvahy	193
14.5	Definice interfejsu	196
14.6	Implementace interfejsu třídou	196
	Anotace <code>@Override</code>	196
	Ověření funkcionality	198
15 Dědění interfejsů		200
15.1	Problém více rolí	200
15.2	Současná implementace více interfejsů	201
	Realizace v kódu	202
15.3	Dědění interfejsů	203
	Trocha teorie o dědění	203
	Dědění a přetypování	204
	Aplikace dědění interfejsů na balíček <code>canvas_3</code> – balíček <code>canvas_4</code>	205
	<code>ICanvasPaintable</code>	206
	<code>IResizable</code> , <code>IMovable</code>	206
	<code>IChangeable</code>	206
	<code>IModular</code>	206
	<code>IShape</code>	207
	<code>MultiShape</code>	207
15.4	Deklarace rodičů interfejsu v kódu	207
15.5	Vlastnosti interfejsů na příkladu <code>MultiShape</code>	207
	Podrobnosti o třídě <code>MultiShape</code>	208
	Mnohotvar se skládá z kopií	208
15.6	Příklad: definice vozidla	209
	Kontrakt	209
	Test vytvořeného vozidla	212
	Problém plátna	213
16 Návrhové vzory		214
16.1	Úvod do návrhových vzorů	214
16.2	Přehled vzorů, s nimiž jsme se již setkali	216
	Knihovní třída (Utility class)	216
	Statická tovární metoda (Static factory method)	216
	Jedináček (Singleton)	217
	Výčtový typ (Enumerated type)	217
	Multi-ton, originál	217
	Služebník (Servant)	218
	Prázdný objekt (Null Object)	218
	Prototyp (Prototype)	219
16.3	Teorie k nové koncepci kreslení	220
	Návrhový vzor Prostředník (Mediator)	220
	Inverze závislostí	221
	Návrhový vzor Pozorovatel (Observer), hollywoodský princip	222

16.4	Správce plátna – CanvasManager	223
	Balíček canvasmanager.....	224
	Import klíčových tříd knihovny	225
16.5	Nová verze třídy Robot	226
17	Prohlubujeme znalosti	229
17.1	Statický import.....	229
17.2	Další členy interfejsů.....	230
	Statické konstanty.....	230
	Statické metody.....	232
	Implicitní metody.....	232
	Soukromé metody.....	233
17.3	Návrhový vzor Adaptér (Adapter)	233
17.4	Návrhový vzor Přepravka.....	234
	Třídy typu záznam (record).....	235
	Záznamy v používané knihovně.....	236
17.5	Návrhový vzor Šablonová metoda	238
	Příklad.....	239
17.6	Návrhový vzor Abstraktní továrna	239
	Motivace	239
	Návrhový vzor Tovární metoda.....	240
	Co je abstraktní továrna.....	240
	Použití.....	240
	Tovární třída převádí konstruktory a statické členy na instanční	241
17.7	Shrnující úloha.....	242
18	Lambda-výrazy, funkční interfejsy a generické typy.....	243
18.1	Nezávislé opakování zadané akce	243
	Možná řešení.....	244
18.2	Syntaxe lambda-výrazů.....	244
	Aplikace na světlo.....	245
18.3	Lambda-výrazy a lokální proměnné.....	247
	Zásobník návratových adres – ZNA	248
18.4	Lambda-výrazy zastupující metody	249
18.5	Funkční interfejsy	250
18.6	Generické datové typy	250
	Motivace	250
	Syntaxe zadávání a používání generických typů	251
	Specifika generických typů Javy	251
	Omezení typových parametrů	252
	Příklad: Interval.....	252
	Typové parametry s více předky.....	252
	Potomci a předci generických typů.....	253
18.7	Funkční interfejsy – pokračování.....	254
	Demonstrace použití.....	255
18.8	Lambda-výrazy nelze přetypovat na Object přímo.....	258
19	Dědění implementace.....	261
19.1	Tři druhy dědění.....	261
	Přirozené (nativní) dědění	261
	Dědění typu	262
	Dědění implementace.....	262
	LSP – Liskov Substitution Principle.....	263
	Shrnutí.....	263
19.2	Základy dědění tříd.....	263
	Univerzální (pra)rodič Object.....	264
	Instance třídy Object jako parametr či návratová hodnota.....	264

19.3	Co dědíme od třídy <code>Object</code>	264
	Class-objekt.....	265
	Úplný název tříd v <code>JShell</code>	266
	Přehled veřejných členů zděděných od třídy <code>Object</code>	266
19.4	Definice třídy s předkem – <code>Square</code>	267
	Rodičovský podobjekt.....	268
	Volání rodičovského konstruktora	269
19.5	Přebíjení metod.....	270
19.6	Skrytá nebezpečí: úprava třídy <code>Square</code>	272
19.7	Virtuální versus konečné metody	274
19.8	Zakrývání statických metod.....	274
	Metody interfejsů se nezakrývají.....	275
19.9	Jediný implementační předek	275
20	Abstraktní třídy	276
20.1	Abstraktní třídy a jejich role v dědicí hierarchii.....	276
	Definice abstraktních tříd a metod.....	278
	<code>public abstract class AX { ... }</code>	278
	<code>public abstract void method();</code>	278
	Experimenty s abstraktní třídou.....	278
20.2	Účel abstraktních tříd	280
20.3	Zavedení abstraktních tříd do projektu	280
20.4	Návrhový vzor <code>Stav</code>	281
20.5	Aplikace na příklad s vozidlem.....	282
	Hlavní třída vozidla – třída <code>Robot4_4</code>	283
	Stavově nezávislé metody	286
	Stavově závislé metody.....	286
	Společný rodič jednosměrných tříd – třída <code>ARobot1_4</code>	286
	Statická tovární metoda	286
	Datové členy a konstruktory	288
	Stavově závislé metody	288
	Jednostavové (jednosměrné) třídy	288
	Test.....	290

Část D Knihovny 293

21	Speciální datové typy.....	294
21.1	Třída <code>Object</code>	294
21.2	Odkazové a hodnotové datové typy	295
21.3	Metody <code>equals(Object)</code> a <code>hashCode()</code>	296
	Metody <code>equals(Object)</code>	296
	Metoda <code>hashCode()</code>	297
21.4	Proměnné a neměnné hodnotové typy.....	297
21.5	Primitivní a obalové typy	299
21.6	Třída <code>String</code>	299
	Možné problémy při práci s některými znaky	300
	Interfejs <code>java.lang.CharSequence</code>	300
	Problémy s kódováním znaků	301
	Třídy <code>StringBuilder</code> a <code>StringBuffer</code>	302
21.7	Výčtové typy.....	303
	Složitější příklad: <code>Direction</code>	304
21.8	Třída <code>Throwable</code>	306
21.9	Třída <code>Thread</code>	306
21.10	Třída <code>java.util.Optional<T></code> & spol.....	306

Motivace	306
Alternativní řešení	307
21.11 Třída <code>java.util.Random</code>	308
<code>int nextInt()</code> <code>long nextLong()</code> <code>int nextInt(int bound)</code>	308
<code>double nextDouble()</code>	308
<code>double nextGaussian()</code>	308
<code>DoubleStream doubles()</code> <code>IntStream ints()</code> <code>LongStream longs()</code>	308
<code>DoubleStream doubles()</code> <code>IntStream ints()</code> <code>LongStream longs()</code>	309
21.12 Interfejs <code>java.lang.Comparable<T></code>	309
21.13 Interfejs <code>java.util.Comparator<T></code>	310
22 Výjimky	312
22.1 Co to jsou výjimky	312
22.2 Nejdůležitější výjimky	313
22.3 Vyhození výjimky	314
Reakce systému na vyhození výjimky	314
22.4 Výjimky a nedosažitelný kód	316
22.5 Hierarchie dědění výjimek	316
22.6 Zachycení vyhozené výjimky	318
22.7 Společný úklid – blok <code>finally</code>	319
22.8 Definice vlastních výjimek	321
22.9 Kontrolované výjimky	322
22.10 Převedení kontrolované výjimky na nekontrolovanou	323
23 Kontejnery	325
23.1 Co je to kontejner v Javě	325
23.2 Kategorizace kontejnerů	326
23.3 Knihovna kolekcí	326
Collections – kolekce	327
Set – množina	328
SortedSet – uspořádaná množina	328
List – seznam	328
Queue – fronta	328
Deque – oboustranná fronta	328
Map – Mapa	329
Collections	329
23.4 Deklarujte typy co nejobecněji	329
23.5 <code>Collection<E></code> – kolekce	330
<code>boolean add(E o)</code>	330
<code>boolean addAll(Collection<? extends E> c)</code>	330
<code>void clear()</code>	330
<code>boolean contains(Object o)</code>	330
<code>boolean containsAll(Collection<?> c)</code>	330
<code>boolean isEmpty()</code>	330
<code>Iterator<E> iterator()</code>	330
<code>boolean remove(Object o)</code>	330
<code>boolean removeAll(Collection<?> c)</code>	331
<code>boolean retainAll(Collection<?> c)</code>	331
<code>int size()</code>	331
23.6 Dvojtečkový cyklus <code>for(:)</code>	331
23.7 Množiny – <code>Set<E></code>	331
23.8 Seznamy – <code>List<E></code>	333
Pořadí prvků	333
Příklad: seřazení prvků v seznamu	335

23.9	Slovníky a mapy – <code>Map<K,V></code>	337
	Pohledy.....	338
	<code>Set<K> keySet()</code>	338
	<code>Collection<V> values()</code>	338
	<code>Set<Map.Entry<K,V>> entrySet()</code>	338
24	Pole.....	339
24.1	Představení.....	339
24.2	Atributy a metody polí.....	340
24.3	Pole jako kontejner.....	340
	Pole odkazů na objekty.....	340
	Pole hodnot primitivních typů.....	341
	Hlídaní mezi polí.....	343
	Inicializace polí v deklaraci.....	343
	Inicializace vytvářeného pole.....	344
	Neinicializovaná pole objektových typů.....	345
24.4	Vícerozměrná pole.....	346
	Obdélníková pole.....	346
	Neobdélníková pole.....	347
	Inicializace vícerozměrného pole.....	348
24.5	Metody s proměnným počtem argumentů.....	348
24.6	Pole, kolekce a moderní programování.....	349
24.7	Závěrečný příklad.....	349
25	Interní datové typy.....	352
25.1	Přehled.....	352
	Terminologie.....	352
	Společné charakteristiky.....	353
	Použití.....	353
	Jedna hladina soukromí.....	354
25.2	Globální interní (členské) datové typy.....	354
25.3	Vnořené datové typy.....	355
25.4	Vnitřní třídy.....	356
25.5	Lokální třídy.....	356
	Pojmenované lokální třídy.....	357
	Anonymní třídy.....	357
25.6	Návrhový vzor Iterátor.....	358
	Interfejsy <code>java.util.Iterator<E></code> a <code>java.lang.Iterable<E></code>	358
	<code>boolean hasNext()</code>	359
	<code>E next()</code>	359
	<code>void remove()</code>	359
	Použitelnost cyklu <code>for(:)</code>	360
	Vnější (sekvenční) a vnitřní (dávkové) iterátory.....	360
26	Datovody.....	361
26.1	Analogie.....	361
26.2	Druhy operací.....	362
26.3	Princip práce datovodu.....	363
26.4	Specifické vlastnosti.....	363
26.5	Datové typy související s datovody.....	364
26.6	Vytváření datovodů.....	364
	Kolekce.....	365
	Pole.....	365
	Interfejs <code>java.util.stream.Stream</code>	366
26.7	Koncepční příklad.....	366
26.8	Operace s daty v datovodu.....	367

Koncové operace.....	367
Průběžné operace.....	367
Částečně průběžné operace.....	368
26.9 Kolektory.....	368
static <T> Collector<T,?,List<T>> toList().....	369
static <T> Collector<T,?,Set<T>> toSet().....	369
static <T,K,U> Collector<T,?,Map<K,U>> toMap(Function<? super T,? extends K> keyMapper, Function<? super T,? extends U> valueMapper).....	369
Příklad.....	369
27 Čtení a ukládání dat	372
27.1 Koncepce čtení a ukládání dat	372
27.2 Soubory: bleskové opakování.....	373
Soubor, souborový systém, cesta.....	373
Relativní, absolutní a kanonická cesta.....	374
Substituované disky ve Windows.....	374
27.3 Třída <code>java.io.File</code>	375
Instanční metody	375
Metody vracející cestu či soubor	375
Zjišťovací metody	376
Vytváření souborů a složek	376
Odstraňování souborů a složek	376
Zjišťování obsahu složek.....	376
27.4 Návrhový vzor Dekorátor	378
Motivace	378
Princip funkce.....	379
27.5 Rozdělení datových proudů.....	380
27.6 Zápis textů.....	381
Splachování a zavírání proudů	382
Přidávání dat na konec existujícího souboru	382
Příklad.....	383
27.7 Čtení textů	385
Příklad.....	386
27.8 Třída <code>java.util.Scanner</code>	387
28 Vytvoření aplikace.....	388
Prohlížení obsahu JAR-souborů.....	388
28.1 JAR-soubor	389
28.2 Demonstrační aplikace.....	389
28.3 Hlavní třída aplikace.....	390
Vytvoření souboru JAR s aplikací	390
28.4 Spuštění aplikace	392
28.5 Soubor <code>MANIFEST_MF</code>	393

Literatura 394

Rejstřík 396

Úvod

Otevíráte čtvrté, zcela přepracované vydání knížky, která vás chce naučit programovat moderním, objektově orientovaným stylem. Stylem, jímž se v dnešní době vyvíjí drtivá většina klíčových aplikací. Oproti předchozím vydáním je kniha od základů přepracovaná. Stejně jako v předchozím vydání je výklad rozdělen do dvou dílů.

První díl (ten právě čtete) se soustředí především na výklad základních syntaktických konstrukcí a architektonických principů, které by si měl čtenář osvojit předtím, než se pustí do kódování složitějších projektů. Jeho cílem je, aby čtenáři přešly tyto principy co nejdříve do krve.

Plánovaný druhý díl pak získané návyky prohloubí a seznámí čtenáře s řadou dalších programátorských technik a prohloubí znalosti standardní knihovny. Postupně se v něm navrhuje jednoduchá, ale na druhou stranu netriviální aplikace. V průběhu tohoto návrhu kniha vysvětluje hlubší souvislosti, na něž v běžných učebnicích již nezbyvá místo, a doplní čtenářovy znalosti zásad návrhu aplikací.

Kniha je výsledkem mnohaletého experimentování s tím, jak co nejlépe učit *soudobé* programování. Vyzkoušeli jsme si výuku programování snad na všech typech zařízení. Učili jsme v zájmových kroužcích na základní škole, na střední škole, na univerzitách i v rozšiřujících kurzech pro profesionální programátory.

Ve svých kurzech neustále zjišťujeme, že střední i vysoké školy opouští řada programátorů, kteří sice programují v nějakém „moderním“ jazyce, ale neumějí programovat moderně. Absolvované kurzy je sice naučily navržený program zakódovat, ale nenaučily je netriviální program samostatně navrhnout. Naučily je používat nejrůznější frameworky, ale oni tyto frameworky používají většinou mechanicky, bez pochopení základních principů, na jejichž základě jsou navrženy.

Řada vysokých škol se v inženýrských oborech snaží naučit své studenty zakódovat program v řadě nejrůznějších jazyků. Neučí je však styl programování, ale především syntaxi a sémantiku probíraných jazyků. Důsledkem jsou pak nestabilní a těžko udržovatelné aplikace.

Ve svých učebnicích se to pokoušíme napravit. Reakce čtenářů prozatím naznačují, že se nám to snad alespoň částečně daří.

Tato učebnice je již podle svého názvu určena pro naprosté začátečníky: máme-li dodržet akceptovatelný rozsah, musíme probrat opravdu jen základy. Pro zájemce proto chystáme učebnici pro mírně pokročilé, která se nesoustředí na kód, ale bude se věnovat spíše návrhu. Představili bychom v ní další zásady moderního programování, včetně např. automatizovaného testování, a v závěru bychom navrhli jednoduchou, avšak netriviální aplikaci od úplného počátku. Vše záleží na zájmu potenciálních čtenářů.

Komu je kniha určena

Tato kniha je určena především těm, kteří ještě nikdy neprogramovali, anebo se je to sice někdo snažil naučit, ale oni už vše zase zapoměli. Knihu jsme se snažili napsat tak, aby ji mohl použít bystrý středoškolák. Nepředpokládá žádné předběžné znalosti a dovednosti kromě základů práce s počítačem. Jejím cílem je předat čtenáři základní znalosti a naučit ho dovednosti potřebné k vytváření jednoduchých aplikací. Osvojené základy mu pak umožní, aby v případě hlubšího zájmu o programování v jazyku *Java* pokračoval některou z učebnic určených pro mírně pokročilé programátory – nejlépe samozřejmě chystaným druhým dílem a referenční příručkou [\[19\]](#) či její následovnicí.

Koncepce výkladu

Musíme vás upozornit na to, že kniha, kterou držíte v ruce, se od běžných učebnic poněkud liší. Ostatní učebnice jsou totiž většinou především učebnicemi nějakého programovacího jazyka. Jejich autoři se proto ve svém výkladu soustředí hlavně na výklad vlastností popisovaného jazyka a jeho knihoven. Bohužel se v nich ale nedozvíte skoro nic o tom, jak při návrhu programů přemýšlet, aby vás nezaskočily náhlé změny zadání, kterými je současné programování pověstné. Takovéto učebnice proto nevychovávají návrháře, kteří by uměli program navrhnout, ale pouze kodéry, kteří umějí zanalyzované zadání zakódovat.



Tady si neodpustíme vzpomínku na jednoho studenta, který na konci semestru vysvětloval, že už profesionálně programuje, takže si myslel, že kurz hravě zvládne. Když jsme na začátku semestru probírali naprosté základy, řekl si, že to je pouhé hraní a že si počká, až začneme doopravdy programovat, a pak chybějící body rychle dožene. Když se k nám však v polovině semestru připojil, zjistil, že řadu konstrukcí, které běžně používáme, vůbec nechápe. Nezbylo nám, než mu zopakovat, že mezi prostým používáním objektových konstrukcí a návrhem objektových programů je velký rozdíl a že řada účastníků přichází do našich kurzů právě proto, aby se tento jiný způsob programátorského myšlení naučila.

Vypadá to, jako když autoři předpokládají, že se při čtení jejich knihy naučíte programovat nějak sami od sebe – obdobně, jako se to museli naučit oni. Zkušenosti s programátory, kteří navštěvují naše kurzy ve firmě či na univerzitě, však ukazují, že tohoto výsledku bývá dosaženo jen zřídka.¹ Většina z nich zná poměrně dobře konstrukce nějakého objektově

¹ Výzkum z přelomu století ukázal, že pouze 10 % programů psaných v objektově orientovaných jazycích je navrženo opravdu objektově. (Goddard, D. 1994. Is it really object oriented? *Data Based Advis.* 12, 12 (Dec. 1994), 120-123.) Od té doby se situace trochu zlepšila, nicméně na svých školeních stále pozorují, že strukturovaně navržené programy psané v objektových jazycích v mnoha softwarových firmách převažují.

orientovaného programovacího jazyka a základy často používaných frameworků. Bohužel, skoro nikdo z nich v něm neumí objektivně programovat, neumí přepnout na objektivní způsob uvažování.

Pro jistotu proto varuji: toto není učebnice jazyka *Java*, toto je učebnice objektivního programování v *Javě*. My se primárně nebudeme učit, jak program zapsat, ale jak jej navrhnout. Cílem není vychovat z čtenářů kodéry v *Javě*, ale připravit je na to, aby z nich mohli vyrůst schopní architekti. Nebudeme vás proto učit specialitám použitého programovacího jazyka (ty jsou podrobně popsány v knize [19]), ale pokusíme se vás naučit efektivně navrhovat a vytvářet spolehlivé a snadno udržovatelné programy. Jinými slovy: chceme vás naučit dovednostem, které budete používat, ať už budete programovat v jakémkoliv objektivně orientovaném jazyku. Jazyky přicházejí a odcházejí. Základní programátorské techniky a způsob myšlení však žijí daleko déle než jazyky, se kterými byly zavedeny.

Díky tomu, že se místo na programovací jazyk soustředíme spíše na vlastní programování, vznikl v knize prostor pro výklad řady programátorských zásad a technik, o nichž se klasické učebnice vůbec nezmiňují, a to často ani učebnice pro zkušené programátory. Tyto zásady a techniky se většinou přednášejí až v nadstavbových kurzech, které učí vytvářet programy tak, aby s vámi mohly růst a aby vás nezaskočily rychle se měnící požadavky zákazníků (a připravte se na to, že tyto měnící se požadavky vás potkají i v případě, kdy jste zákazníkem sami sobě). Přitom vůbec nejde o techniky složité, které by začátečník nezvládl pochopit. Ostatní učebnice je pomíjejí pouze proto, že nesouvisejí přímo se syntaxí programovacího jazyka, ale týkají se obecného programování.

Tato učebnice je naopak vysvětluje od samého počátku výkladu, protože víme, že byste si je měli osvojit co nejdříve. Ti, kteří se nejprve učí kódovat, a teprve následně se dozvídají, jak lze návrh programu zefektivnit, bývají příliš v zajetí svých zkušeností s kódováním a při návrhu programů se pak zbytečně silně soustředí na některé nepodstatné detaily.

Potřebné vybavení

K vývoji programů budete potřebovat vývojovou sadu JDK, kterou můžete stáhnout na adrese <https://www.oracle.com/java/technologies/javase-downloads.html>. Potřebujete ale sadu JDK 17 nebo mladší. Na starších verzích *Javy* by naše programy nemusely pracovat. Počítejte s tím, že instalační soubory zabírají zhruba 160 MB a po instalaci bude sada zabírat přes 300 MB.

K vývojové sadě je vhodné si stáhnout a nainstalovat i dokumentaci. Pokud ale neopouštíte web, tak můžete využívat i její trvalou instalaci na webu. Musíte se však smířit s tím, že ji seženete pouze anglicky. ZIP soubor s dokumentací zabírá asi 50 MB a po rozbalení bude zabírat zhruba 460 MB.