KAPITOLA 5

Jak javové programy fungují

V této kapitole:

- Dozvíte se, jak pracují aplikace
- Naučíte se strukturovat svoje aplikace
- Ukážeme vám, jak se aplikaci předávají argumenty
- Dozvíte se, jak jsou javové programy obvykle uspořádány
- Seznámíte se s možnostmi využívání knihovny tříd Java Class Library (JCL)
- Vytvoříte ve své aplikaci objekt

Poměrně důležitý rozdíl mezi javovými programy je dán cílovým prostředím, v němž mají být tyto programy provozovány. Některé programy jsou totiž navrženy tak, aby byly spouštěny na počítači. Jiné jsou zase připravovány tak, aby běžely jako součást webové stránky.

Javové programy určené pro běh na lokálním počítači se nazývají *aplikacemi*. Naopak programy běžící v rámci nějaké webové stránky ve webovém prohlížeči se nazývají *aplety*, zatímco programy spouštěné webovými servery se nazývají *servlety*. V případě programů pro mobilní zařízení pak hovoříme někdy o tzv. *apkách*.

Programy, které budete vytvářet v této knize, se označují pojmem *mody Minecraftu* a běží v prostředí serveru Minecraftu. Kód samotného serveru je napsán v Javě, díky čemuž tyto programy fungují vlastně jako rozšíření tohoto serveru.

V této kapitole vytvoříte svoji další aplikaci a spustíte ji na svém počítači.

Vytvoření aplikace

Mod Minecraftu, který jste napsali během studia kapitoly 3 „Vytvoření modu pro Minecraft“, je příkladem javové aplikace. Aplikace, kterou začnete vytvářet níže, bude o něco jednodušší, neboť se nebude jednat o mod.

Aplikace DruhaOdmocnina vypočítá druhou odmocninu čísla a zobrazí výsledek.

Máte-li v prostředí NetBeans stále otevřený projekt Minecraft, můžete zahájit práci na nové aplikaci:

1. Z nabídky File zvolte New File. Spustí se průvodce New File.
2. V levé části tohoto průvodce vyberte kategorii Java. Poté přejděte do pravé části dialogu, v níž zvolte Empty Java File. Pokračujte klepnutím na tlačítko Next.
3. Do pole Class Name zadejte název třídy `DruhaOdmocnina`.
4. Do pole Package zadejte název balíčku `com.javaminecraft`.
5. Dialog ukončete klepnutím na tlačítko Finish.

Prostředí NetBeans vytvoří soubor zdrojového kódu `DruhaOdmocnina.java` a otevře jej v okně editoru zdrojového kódu. Díky tomu můžete na kódu své nové aplikace hned začít pracovat. Do editoru zadejte vše, co vidíte ve výpise 5.1, přičemž nezapomeňte vynechat čísla řádků a za nimi následující dvojtečky, nacházející se na začátku každého řádku (připomeňme si, že čísla uvádíme ve výpisech jenom proto, abychom později mohli kód snáze popisovat). Po dokončení uložte všechny provedené změny klepnutím na tlačítko Save All, nacházející se na panelu nástrojů.

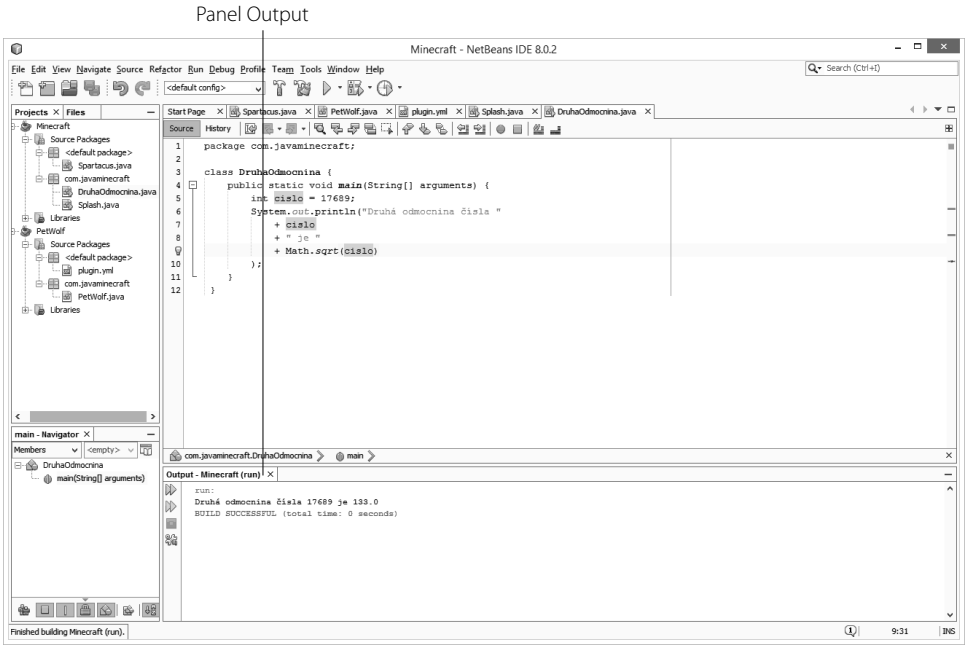
Výpis 5.1: Úplný zdrojový kód programu DruhaOdmocnina.java

```
1: package com.javaminecraft;
2:
3: class DruhaOdmocnina {
4:     public static void main(String[] arguments) {
5:         int cislo = 17689;
6:         System.out.println("Druhá odmocnina čísla "
7:             + cislo
8:             + " je "
9:             + Math.sqrt(cislo)
10:        );
11:    }
12: }
```

Aplikace DruhaOdmocnina provádí následující úkony:

- Řádek 1: Aplikace je přidána do balíčku `com.javaminecraft`.
- Řádek 5: Číslo 17 689 je uloženo do proměnné s názvem `cislo`.
- Řádky 6-10: Aplikace nakonec zobrazí nejen toto číslo, ale i jeho druhou odmocninu. K zobrazení druhé odmocniny slouží příkaz `Math.sqrt(cislo)` na řádku 9.

Pokud jste zdrojový kód uvedený ve výpise 5.1 zadali bez jakýchkoliv překlepů či dalších chyb, a to včetně veškerých interpunkčních znamének a s dodržením velkých a malých písmen dle našeho vzoru, můžete v prostředí NetBeans přejít do nabídky Run, z ní zvolit Run File, a program tak rovnou spustit. Ve spodní části prostředí NetBeans se zobrazí panel Output, v němž uvidíte výstup svého dalšího programu. Ukázku vidíte na obrázku 5.1.



Obrázek 5.1: Výstup aplikace DruhaOdmocnina

Spustíte-li jakoukoliv javovou aplikaci, virtuální stroj Javy (*Java Virtual Machine – JVM*) nejprve vyhledá blok kódu `main()` a poté začne provádět veškeré příkazy v něm uvedené. Pokud však takový blok kódu nenalezne, JVM zobrazí chybové hlášení.

Příkaz `Math.sqrt(cislo)` na řádku 9 názorně ukazuje jednu z vestavěných schopností jazyka Java – v tomto případě schopnost vypočítat druhou odmocninu nějakého čísla. Stručně řečeno, součástí standardního javového programu s názvem `Math` je metoda pojmenovaná `sqrt()`, která dokáže vypočítat druhou odmocninu zadaného čísla.

Program `Math` je součástí knihovny tříd Java Class Library, jíž se budeme zabývat dále v této kapitole.

Předávání argumentů aplikacím

Javové aplikace můžete také spouštět z příkazového řádku, a to pomocí programu `java`, spouštějícího JVM. Prostředí NetBeans využívá tento program v pozadí, ve chvíli, kdy spouštíte nějaký program. Je-li javový program spouštěn jako příkaz, aplikaci načítá JVM. Součástí příkazu pak mohou být i nějaké další informace, jako například v tomto příkladu:

```
java DataEditor game.dat delta
```

Veškeré dodatečné informace předávané programu se pak nazývají *argumenty*. První argument, je-li tedy vůbec zadán, následuje vždy za první mezerou umístěnou za názvem aplikace.

Každý další argument je pak od toho předcházejícího oddělen také jednou mezerou. V předcházejícím příkladu tedy DataEditor představuje název aplikace, zatímco `game.dat` a `delta` jsou argumenty.

Chcete-li vložit mezeru přímo do argumentu, pak musíte daný argument uzavřít uvozovkami. Ukázku vidíte na následujícím řádku:

```
java DataEditor game.dat delta "Nova Hra"
```

Tento ukázkový příkaz by spustil program DataEditor se třemi argumenty: `game.dat`, `delta` a `"Nova Hra"`. Díky uvozovkám totiž nebudou slova `Nova` a `Hra` považována za samostatné argumenty.

Poznámka: Všechny argumenty předávané javové aplikaci jsou ukládány do řetězců, a to i v případě, že argument je číselný. Chcete-li tedy některý z těchto argumentů použít ve své aplikaci jako číslo, musíte jeho hodnotu zkonvertovat. Více informací o tomto tématu najdete v kapitole 12„Popsání objektu".

Javové aplikaci můžete předat tolik argumentů, kolik chcete (samozřejmě v rozumných mezích). Chcete-li je však nějakým způsobem využít, musíte do své aplikace přidat příkazy, které s nimi dále pracují.

Abychom si ukázali, jak argumenty v aplikaci fungují, vytvoříme si v projektu Minecraft nový program:

1. Z nabídky File zvolte New File.
2. V levé části průvodce New File vyberte kategorii Java. Poté přejděte do pravé části tohoto dialogu, v níž zvolte Empty Java File. Pokračujte klepnutím na tlačítko Next.
3. Do pole Class Name zadejte název třídy `Prikaz`. Do pole Package pak zadejte název balíčku `com.javaminecraft`. Dialog ukončete klepnutím na tlačítko Finish.

Do okna editoru zdrojového kódu zadejte kód uvedený ve výpise 5.2. Po dokončení kód uložte. Následně program zkompilujte a v případě potřeby opravte veškeré chyby zvýrazněné editorem.

Výpis 5.2: Úplný zdrojový kód programu Prikaz.java

```
1: package com.javaminecraft;
2:
3: class Prikaz {
4:     public static void main(String[] arguments) {
5:         System.out.println("Přivolali jste " + arguments[0]
6:         + " " + arguments[1] + " mobů."
7:         );
8:     }
9: }
```

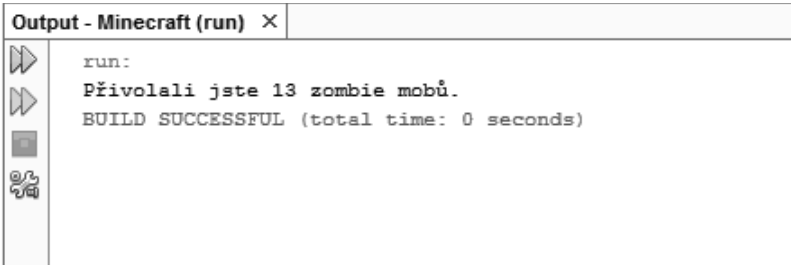
Tuto aplikaci lze normálně zkompilovat a spustit, avšak pokud se o to pokusíte prostřednictvím volby Run File z nabídky Run, zobrazí se vám poměrně nesrozumitelná chyba:

```
Exception in thread "main" java.lang.ArrayIndexOutOfBoundsException: 0
        at com.javaminecraft.Prikaz.main(Prikaz.java:5)
```

Tato chyba je vyvolána proto, že program očekává, že v okamžiku spuštění mu budou předány dva argumenty. V prostředí NetBeans můžete argumenty zadat prostřednictvím úprav projektu:

1. Z nabídky Run postupně zvolte Set Project Configuration a poté Customize. Otevře se dialog Project Properties.
2. Do textového pole Main Class zadejte `com.javaminecraft.Prikaz`.
3. Do pole Arguments zadejte `13 zombie` a poté klepněte na tlačítko OK.

Jelikož jste upravili vlastnosti projektu, musíte jej nyní spouštět poněkud odlišným způsobem. Přejděte do nabídky Run a vyberte v ní Run Project. Z obrázku 5.2 je patrné, že aplikace při svém spuštění použije argumenty, které jste zadali do vlastností projektu:



Obrázek 5.2: Ukázka výstupu aplikace `Prikaz`

Vraťte se se zpět do dialogu Project Properties, číslo `13` změňte na nějaké jiné a namísto moba typu `zombie` zadejte jiný typ moba ze hry Minecraft. Poté projekt znovu spusťte.

Argumenty představují poměrně jednoduchý způsob úprav chování programu. Technicky vzato jsou argumenty ukládány do proměnné typu *pole*. Více informací o polích se dozvíte v kapitole 10 „Ukládání informací do polí".

Mody Minecraftu často pracují s jedním či více argumenty, díky čemuž se s argumenty opět brzy setkáte. Hráči mající na serveru Minecraftu oprávnění operátora mohou využít vestavěný příkaz `Summon` k přidání dalších mobů do hry. Tento příkaz pracuje se třemi argumenty, z nichž prvním je typ moba, druhým jsou souřadnice x, y a z toho místa, v němž by se mob měl objevit, a třetím datové návěští pro dodatečnou konfiguraci. Příklad syntaxe tohoto příkazu vidíte na následujícím řádku:

```
/Summon Bat 206 35 101
```

Tímto přivoláte netopýra na pozici se souřadnicemi (206, 35, 101). Argument datového návěští byl v tomto případě vynechán, což je naprosto v pořádku, neboť se jedná o volitelný argument. Stejně tak je volitelný i druhý argument, popisující pozici. To znamená, že jediným povinným argumentem příkazu `Summon` je první argument, tedy typ moba. Níže vidíte další funkční ukázku příkazu `Summon`:

```
/Summon PigZombie
```

Tímto příkazem přivoláte moba typu zombie pigman na vaše vlastní souřadnice (x, y, z).

Tyto argumenty fungují tedy zcela stejně jako argumenty příkazového řádku javových aplikací.

Knihovna tříd Java Class Library (JCL)

V první polovině této knihy vám vysvětlujeme, jak máte využívat programovací jazyk Java při vytváření vlastních aplikací od samého počátku. Naučíte se v ní všechna klíčová slova a operátory tvořící základ jazyka, a poté je využijete při psaní příkazů, díky nimž bude váš počítač provádět různé zajímavé a užitečné věci.

Ačkoliv tento přístup je zcela určitě nejlepším přístupem pro výuku jazyka Java, lze říci, že se do jisté míry podobá tomu, jako kdybychom někomu vysvětlovali stavbu automobilu tak, že bychom ho nutili, aby si nejdříve sám vyrobil každou potřebnou součástku.

Avšak jestliže již máte nějaké znalosti týkající se Javy, pak víte, že spousta práce je již udělána za vás, pokud ovšem víte, kde ji hledat.

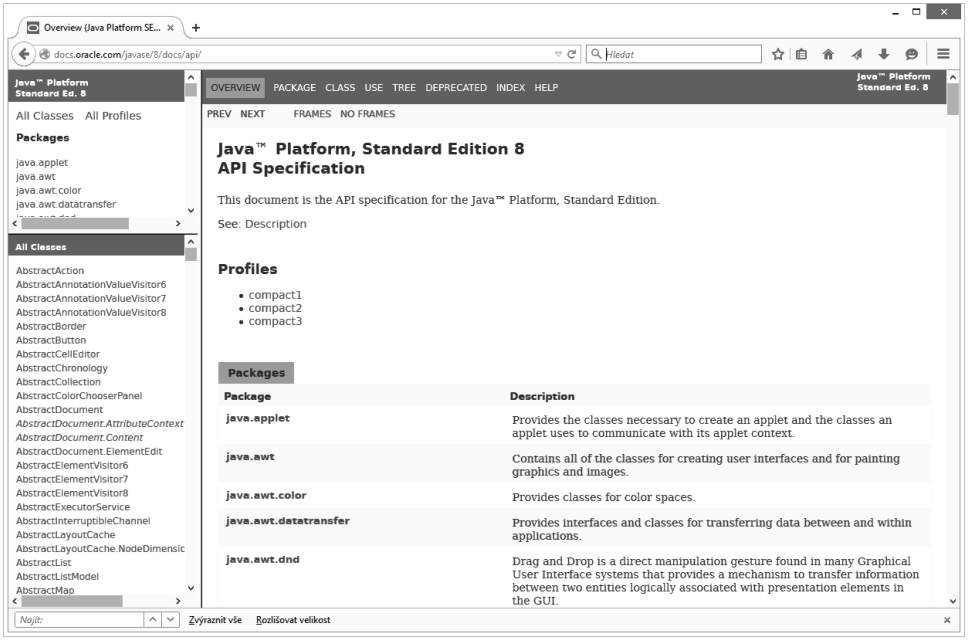
Jazyk Java je dodáván s rozsáhlou kolekcí kódu, zvanou knihovna tříd Javy (Java Class Library – JCL). Tu můžete ve svých programech samozřejmě využít. Tato knihovna je kolekcí tisíců tříd, z nichž mnohé budou pro vás při psaní vlastních programů bezesporu užitečné.

Poznámka: Existuje mnoho dalších knihoven tříd, dodávaných jinými společnostmi a organizacemi. Tak například The Apache Software Foundation, což jsou tvůrci webového serveru Apache, nabízí více než tucet javových projektů s volně přístupným kódem. Jedním z nich je například HttpComponents, sada tříd pro vytváření webových serverů, klientů a crawlerů.
Více informací o tomto projektu najdete na stránce *http://hc.apache.org*. Pokud vás zajímají všechny javové projekty The Apache Software Foundation, pak se podívejte na stránku *http://projects.apache.org*.

Třídy můžete ve svých programech využít způsobem připomínajícím používání proměnných.

Třída umožňuje vytvoření objektu, který lze přirovnat k podstatně zdokonalené a složitější proměnné. Objekt může totiž nejenom uchovávat data, tedy podobně jako proměnná, ale navíc může provádět i různé úkony, čímž se zase podobá programu.

Na adrese *http://docs.oracle.com/javase/8/docs/api* najdete podrobnou dokumentaci společnosti Oracle, popisující knihovnu tříd jazyka Java. Uvedená stránka je zobrazena na obrázku 5.3.



Obrázek 5.3: Dokumentace ke knihovně tříd Java Class Library

Javové třídy jsou uspořádány do balíčků sloužících podobnému účelu jako složka pro ukládání souborů na počítači. Programy, které jste dosud vytvořili, patří do balíčku `com.javaminecraft`.

Úvodní stránka této dokumentace je rozdělena do rámců. V největším z nich najdete přehled všech balíčků, z nichž se skládá celá knihovna tříd Java Class Library, spolu s popisem každého z nich.

Názvy jednotlivých balíčků napovídají, jaký je jejich účel. Tak například balíček `java.io` obsahuje sadu tříd zajišťujících vstup a výstup z pevných disků, internetových serverů a dalších typů datových zdrojů; `java.time` obsahuje třídy související s prací s datem a časem; a v `java.util` najdete mnoho různých pomocných tříd.

Jak jsme si již řekli, v největším rámci domovské stránky dokumentace vidíte seznam všech balíčků spolu s popisem každého z nich. Chcete-li se o některém z balíčků dozvědět více informací, klepněte na jeho název. Přejdete na stránku, na níž uvidíte seznam všech tříd obsažených ve vybraném balíčku.

Platí, že každé třídě, která je součástí knihovny tříd Java Class Library, je na tomto webu s dokumentací věnována samostatná stránka. To znamená, že celý web je dnes tvořen více než 26 000 stránek. (V tuto chvíli po vás rozhodně nechceme, abyste se je snažili všechny přečíst.)

Při přípravě závěrečného projektu této kapitoly se v knihovně tříd Javy jenom trochu porozhlédnete a jednu z existujících tříd využijete k tomu, aby pro vás udělala něco užitečného.

Program `Kostka` využívá třídu `Math` z balíčku `java.lang`. Pomocí této třídy můžete vytvářet náhodná čísla, což je při programování modů Minecraftu poměrně častý požadavek.

V našem programu využijeme metodu `random()` této třídy k vygenerování náhodné hodnoty v rozmezí 1 až 20. Níže vidíte první krok celého procesu:

```
double hod = Math.random() * 20;
```

Metoda `random()` standardně vrací náhodně vygenerované číslo v rozmezí 0,0 až 1,0, přičemž však vygenerované číslo nikdy nemůže být rovno 1,0. Jedná se o číslo s plovoucí desetinnou čárkou, a proto musí být uloženo do proměnné, do níž je možné taková čísla ukládat – v našem případě tedy do proměnné typu `double` s názvem `hod`.

Jelikož je náhodně generované číslo násobeno 20, výsledek se bude pohybovat v rozmezí 0 až 20 (přičemž nikdy nebude roven 20).

Závěrečným krokem při generování takového čísla je jeho zaokrouhlení na nejbližší nižší celé číslo a přičtení hodnoty 1:

```
hod = Math.floor(hod) + 1;
```

Tento příkaz využívá další poměrně užitečnou metodu třídy `Math`. Metoda `floor()` převezme číslo s plovoucí desetinnou čárkou a zaokrouhlí jej směrem dolů. Pokud by tedy proměnná `hod` na počátku obsahovala hodnotu 17,38, tato metoda by vrátila 17,0. A jelikož je vzápětí přičítána hodnota 1, je výsledná hodnota proměnné `hod` rovna 18,0.

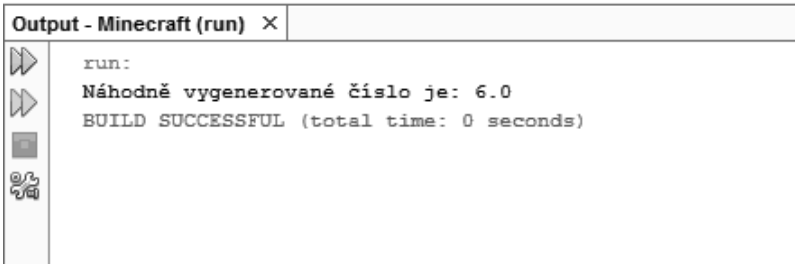
Pokud byste nepoužili třídu `Math` knihovny tříd Java Class Library, museli byste si napsat vlastní program vytvářející náhodná čísla, což je však poměrně náročný úkol. Náhodná čísla jsou užitečná nejen ve hrách, ale i ve výukových a dalších programech, v nichž je nutné provést něco náhodně.

V prostředí NetBeans vytvořte nový prázdný soubor zdrojového kódu v jazyce Java, pojmenujete jej `Kostka` a přidejte jej do balíčku `com.javaminecraft`. Jakmile se vám otevře editor zdrojového kódu, vložte do něj veškerý kód uvedený ve výpise 5.3. Poté klepněte na tlačítko Save All (případně z nabídky File vyberte volbu Save).

Výpis 5.3: Úplný zdrojový kód programu Kostka.java

```
1: package com.javaminecraft;
2:
3: import java.util.*;
4:
5: class Kostka {
6:     public static void main(String[] arguments) {
7:         double hod = Math.random() * 20;
8:         hod = Math.floor(hod) + 1;
9:         System.out.println("Náhodně vygenerované číslo je: " + hod);
10:    }
11:}
```

Nyní přejděte do nabídky Run a program spusťte volbou Run File. Ukázku výstupu vidíte na obrázku 5.4 (je však nutné zdůraznit, že číslo, které se zobrazí vám, bude zřejmě jiné).



Obrázek 5.4: Ukázka výstupu programu `Kostka`

Zkuste si program spustit několikrát po sobě, a to stisknutím klávesové kombinace Shift + F6. Uvidíte, že výsledné číslo se bude vždy pohybovat v rozmezí 1 až 20. Vrátíte-li se do zdrojového kódu a změníte-li na řádku 7 hodnotu 20 na 6, bude se výsledné číslo vždy pohybovat v rozmezí 1 až 6.

Třída `Math` je – podobně jako všechny ostatní třídy knihovny Java Class Library – velice podrobně zdokumentována, přičemž veškerou dokumentaci najdete na výše zmíněném webu společnosti Oracle. Tato dokumentace obecně popisuje smysl třídy, uvádí balíček, do nějž patří, říká, jakým způsobem se vytvářejí objekty této třídy, a vyjmenovává všechny metody, které jsou v ní obsažené a které lze volat, potřebujete-li provést něco užitečného.

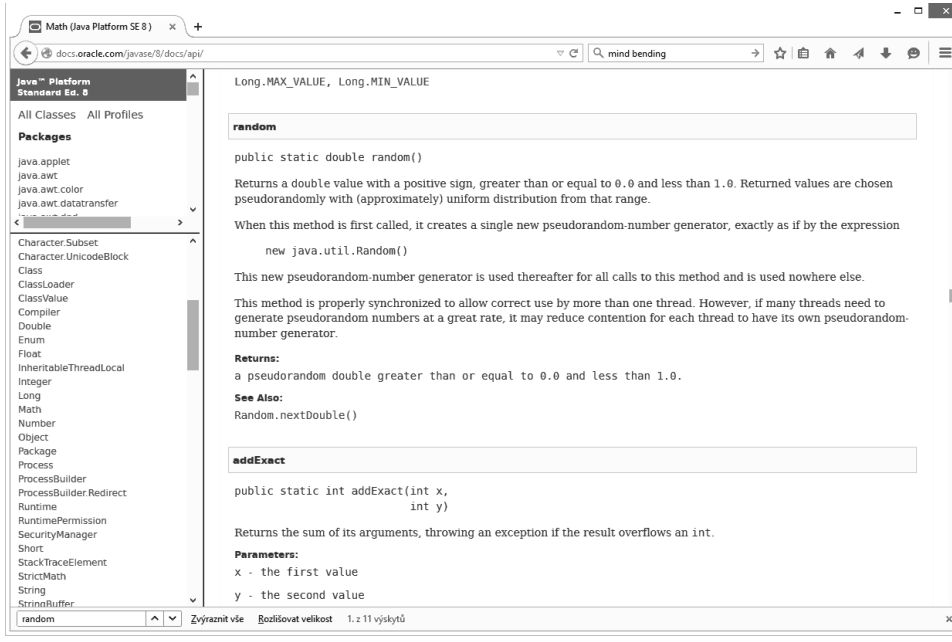
Chcete-li si tuto dokumentaci prohlédnout, pak postupujte takto:

1. Spusťte si webový prohlížeč a přejděte na stránku *http://docs.oracle.com/javase/8/docs/api*.
2. V hlavním rámci se přesuňte směrem dolů, dokud neuvidíte odkaz na balíček s názvem `java.lang`. Klepněte na tento odkaz. Přejdete tak na dokumentaci týkající se vybraného balíčku.
3. V hlavním rámci se přesuňte dolů, dokud nenajdete odkaz na třídu `Math`.

Jako programátor, který se programování v jazyce Java teprve učí, zřejmě zjistíte, že tato dokumentace je pro vás téměř nesrozumitelná. Nemusíte se však ničeho bát: to je dáno tím, že dokumentace je napsána pro již zkušenější programátory.

Může se ale stát, že čtení dalších kapitol ve vás bude vyvolávat zvědavost týkající se způsobu využívání některých vestavěných tříd Javy. V takovém případě by přečtení oficiální dokumentace dané třídy pro vás mohlo být přínosné. Jedním z možných způsobů využití této dokumentace je také prohlédnutí všech metod dané třídy, z nichž každá slouží nějakému určitému účelu.

Na stránce dokumentace ke třídě `Math` můžete přejít dolů a vyhledat si podrobnější popis metody `random()`. Ukázku této části dokumentace vidíte i na obrázku 5.5.



Obrázek 5.5: Dokumentace společnosti Oracle popisující třídu Math

Poznámka: Všechny třídy jazyka Java použité v této knize jsou popsány na výše uvedených stránkách společnosti Oracle. Studium této online dokumentace není pro čtení dalších kapitol a pro vaši „přeměnu" v programátora pracujícího v jazyce Java nezbytné.

Na druhou stranu ale platí, že každá ze tříd použitých v této knize nabízí mnohem více funkcí či možností, než je možné popsat v knize pro začátečníky. Z tohoto důvodu by oficiální dokumentace knihovny tříd Java Class Library mohla být vhodným doplňkem ke studiu této knihy.

Kterákoliv knihovna tříd jazyka Java, s níž budete pracovat, bude mít dokumentaci uspořádanou podobným způsobem jako knihovna tříd Java Class Library. I Bukkit, což je sada tříd umožňujících vytváření modů pro Minecraft, má velice obdobnou online dokumentaci.

Zatím ji sice ještě nepotřebujete, ale jste-li zvědaví a chcete-li se na ni alespoň v rychlosti podívat, navštivte stránku *http://www.javaminecraft.com/bukkitapi*, na níž najdete dokumentaci té knihovny tříd projektu Bukkit, kterou jste si nainstalovali na svůj počítač.

Absolutní základ

Při čtení této kapitoly jste měli příležitost vytvořit si vlastní program v jazyce Java, předat argumenty spouštěnému programu a využít existující programy tvořící součást knihovny tříd Java Class Library (JCL).

V následujících několika kapitolách se budeme dále věnovat aplikacím, přičemž vaše zkušenosti programátora pracujícího v jazyce Java budou nadále vzrůstat. Testování aplikací je rychlejší, neboť jejich spuštění nevyžaduje žádné další speciální úkony, jako je tomu v případě jiných typů javových programů.

Tato kapitola je první z těch, které se zabývají možnostmi využití objektů v javovém programu. K tomuto tématu se vrátíme opět až v kapitole 11, nazvané „Vytvoření prvního objektu“.