

# Co se děje v Javě

Petr Adámek

Fakulta informatiky, Masarykova univerzita, Brno

Email: `petr.adamek@bilysklep.cz`

**Abstrakt:** Přednáška se věnuje aktuálnímu dění v oblasti jazyka Java a je koncipována tak, aby byla přínosem nejen pro stávající či budoucí vývojáře Java aplikací, ale i pro širokou odbornou veřejnost. Kromě technických a technologických prvků se věnuje i otázkám filozofickým a vztahu platformy Java ke světu svobodného a open source software.

*Klíčová slova:* Java, Java 2, JVM, garbage collector, J2EE, JSP, Tomcat, JBoss.

## 1 Úvod

Jazyk Java<sup>1</sup> patří mezi oblíbené nástroje pro tvorbu širokého spektra aplikací, od programů a her pro mobilní telefony, přes GUI aplikace, až po podnikové informační systémy. Neustále prochází bouřlivým vývojem a okolo něj se odehrává mnoho zajímavých věcí. Tento příspěvek volně navazuje na přednášku *Java a Linux* z konference SLT2004 a snaží se čtenáři zprostředkovat ty nejzajímavější a nejvýznamnější události, které se v oblasti platformy Java odehrály v poslední době nebo které se v brzké době odehrají.

Tento přehled není (a ani nemůže být) úplný, nicméně doufám, že jsem na nic důležitého nezapomněl a že si v něm každý čtenář najde něco svého.

## 2 Java a Open Source

Někteří skalní zastánci myšlenek svobodného (Free) software kritizují fakt, že Java je plně pod kontrolou firmy SUN Microsystems a že není k dispozici pod nějakou rozumnou (čti schválenou OSI) svobodnou licenci. Dokonce i firma IBM se dlouhodobě snaží přimět SUN k ještě většímu otevření vývoje Javy, zatímco SUN Microsystems toto neustále zvažuje a prozatím nedospěla k definitivnímu rozhodnutí. Takže v tomto směru nedošlo k žádné změně a postoj firmy SUN Microsystems je zatím spíše odmítavý.

Komunita vývojářů je rozdělena na dva tábory. Zastánci úplného otevření vývoje Javy pod nějakou free licenci argumentují klasickými argumenty o výhodách tohoto přístupu, zatímco odpůrci tohoto kroku poukazují na riziko přílišné diverzifikace jednotlivých větví vývoje a ztráty klíčové výhody platformy Java, kterou je přísná standardizace a plná přenositelnost.

---

<sup>1</sup> `java.sun.com`

Osobně se domnívám, že prozatím není současná licence Javy zásadní problém – vývoj je docela otevřený, každý může přispět do procesu JCP, jsou k dispozici zdrojové kódy k většině nástrojů i knihoven a Javu je možné využívat i šířit téměř bez omezení. Kdokoliv může vytvořit alternativní implementaci, pouze ji bez příslušné certifikace nesmí označovat názvem a logem Java.

Problém by mohl nastat tehdy, kdyby se firma SUN dostala do problémů nebo se jen rozhodla změnit svoji obchodní politiku. Nicméně si myslím, že komunita vývojářů v čele s firmami jako IBM či Oracle je poměrně silná a s takovou situací by si poradila. Stejně tak se neobávám nedávného smíření firem SUN Microsystems a Microsoft. Viděl bych v tom ryze pragmatický krok obou společností, jehož cílem je eliminovat náklady na právní zastoupení a riziko prohrání vzájemných soudních sporů.

Co se týče open source implementací Javy, situace se začíná pomalu obracet k lepšímu a vývojáři projektů GJC<sup>2</sup> a Classpath<sup>3</sup> odvedli pořádný kus práce; přesto však jejich projekty bohužel dost zaostávají za oficiálními implementacemi dodávanými firmami SUN Microsystems a IBM. Zde se dle mého názoru projevuje fakt, že díky licenci a otevřenosti oficiálních implementací nejsou vývojáři nuceni používat jejich open source varianty, takže do nich ani nepřispívají a celkově je množství lidí pracujících na nich menší, než jsou armády vývojářů u komerčních firem.

Přes všechny problémy s licencí je jazyk Java velmi oblíben i mezi vývojáři open source a svobodného software a množství projektů využívajících tento jazyk neustále roste.

### 3 Java a .NET

Aniž bych chtěl snižovat význam různých alternativních platforem (postavených např. na jazycích Python, Perl, Smalltalk či C/C++), v současné době je v podstatě jediným plnohodnotným soupeřem platformy Java platforma .NET od firmy Microsoft. Ačkoliv bylo původním záměrem této firmy vyvinout systém, který bude výrazně lepší než Java, po hlubším srovnání zjistíte, že klíčové principy a myšlenky jsou naprosto totožné. Rozdíl spočívá pouze v přístupu a v technických detailech.

#### 3.1 Co je momentálně lepší?

Nemyslím si, že by v současné době byla Java výrazně lepší než .NET nebo naopak. Osobně však upřednostňuji Javu, a to z několika důvodů:

- Java je plně přenositelná mezi mnoha platformami;
- Java se mi jeví jako více otevřená a na jejím vývoji se podílí široká komunita vývojářů v rámci projektu JCP;

<sup>2</sup> <http://gcc.gnu.org/java/>

<sup>3</sup> [www.gnu.org/software/classpath/](http://www.gnu.org/software/classpath/)

- Java je podporována více firmami (SUN Microsystem, IBM, Oracle, Borland, HP atd.);
- pro Javu existuje velké množství volně dostupných nástrojů a knihoven;
- Java je čistý a dobře navržený jazyk, který se brání různým kompromisům a špinavým trikům.

Na druhou stranu spatřuji i některé výhody platformy .NET:

- díky pozdějšímu uvedení na trh se mohla vyhnout některým nedostatkům platformy Java, které se v průběhu jejího vývoje objevily;
- .NET Visual Studio je velmi mocný nástroj, umožňující snadný WYSWYG vývoj internetových aplikací.

Poslední výhodu si velmi dobře uvědomuje i firma SUN Microsystems, která proto vydává produkt Sun Java Studio Creator, který má v tomto směru Visual Studiu od firmy Microsoft konkurovat. A rozhodně půjde o silného konkurenta.

### 3.2 Mono

Projekt Mono<sup>4</sup> je implementací .NET pro Linux. Prochází poměrně intenzivním vývojem a jeho vývojáři mají velmi ambiciózní plány.

Ačkoliv tomuto projektu velice fandím, stavím se k němu poněkud skepticky. Jednak není stále v příliš použitelném stavu, a navíc se obávám, že by v případě jeho úspěchu firma Microsoft mohla šikovně využít patentové ochrany některých částí platformy .NET, aby projekt Mono udržela v pro ni přijatelných mezích<sup>5</sup>.

### 3.3 Shrnutí

Co se týče souboje platforem .NET a Java, zatím nemá žádného vítěze ani poraženého; firmy SUN Microsystems ani Microsoft zatím nevysypaly z rukávu nějaké eso, kterým by převálcovaly konkurenci.

V každém případě se domnívám, že je pro programátory existence alternativ a konkurence velmi užitečná, a byl bych velmi nerad, kdyby Java nebo .NET začaly výrazně zaostávat nebo dokonce ukončily svůj vývoj. Pokud vývojáři dodržují standardy a používají známé a otevřené prostředky pro komunikaci, není problém s interoperabilitou a každý projekt může použít takové řešení, které je pro něj nejvýhodnější.

## 4 J2SE 1.5

Jednou z největších událostí letošního roku je zcela jistě příchod nové verze jazyka Java. Tato verze s číslem 1.5 je označována za významný milník ve vývoji tohoto jazyka a někteří zasvěcení prohlašují, že je to ještě větší zlom než příchod

<sup>4</sup> [www.go-mono.com](http://www.go-mono.com)

<sup>5</sup> Tj. aby se dalo hovořit o existenci alternativ, ale aby nebyly příliš použitelné a všichni zákazníci si raději vybrali originální řešení od firmy Microsoft.

revoluční verze 1.2 v roce 1998<sup>6</sup>. Na ty nejvýznamnější z nich se podíváme. Další informace lze nalézt např. v [4], [2] nebo [1].<sup>7</sup>

Verze 1.5 platformy Java má být firmou SUN Microsystems uvolněna letos v létě, nicméně je již nyní možné stáhnout její betaverzi<sup>8</sup>. Pokud byste si chtěli níže popsané konstrukce vyzkoušet, nezapomeňte překladač spouštět s parametrem `-source 1.5`. Jinak bude překladač nové konstrukce považovat za syntaktickou chybu.

#### 4.1 Efektivita

Mezi pravidelné změny v každé nové verzi jazyka Java patří zvýšení rychlosti; v tomto směru Java 1.5 nikterak nevybočuje. Kromě několika změn v garbage collectoru a virtuálním stroji je asi nejvýznamnějším prvkem vlastnost nazvaná *class data sharing*. Tento mechanismus urychluje start virtuálního stroje a šetří paměť v případech několika souběžně spuštěných virtuálních strojů. Princip spočívá v tom, že se při instalaci JRE (nebo Java SDK) spustí virtuální stroj a pak je paměťová oblast vyhrazená pro read-only data uložena do souboru na disk. Při každém dalším spuštění JVM se tato oblast namapuje do paměti, což start výrazně urychlí. Pokud běží více virtuálních strojů naráz, tato oblast je sdílená.

Mechanismus funguje pouze pro klientskou verzi JVM; serverová verze podporována není, což je ale pochopitelné. Serverové aplikace jsou většinou vytvářeny jako vícevláknové v rámci jednoho virtuálního stroje a běží nepřetržitě.

#### 4.2 Generické datové typy

Jazyk Java má silnou typovou kontrolu, nicméně v některých případech ji není možné plně využít a je nutné používat přetypování. Typickým příkladem je knihovna kolekcí. Aby bylo možné do standardních kolekcí v Javě ukládat libovolný objekt, pracují tyto kolekce s obecným typem `Object`. Pokud chceme do kolekce ukládat objekty konkrétního typu, neexistuje jednoduchý mechanismus, který by znemožnil do této kolekce vložit i objekt jiného typu. Při výběru objektu z kolekce je nutné jej navíc explicitně přetypovat. Typický kód pro práci s kolekcí pak vypadá takto:

```
List list = new ArrayList();
// ...
list.add(new Integer(1));
// ...
for (Iterator i=list.iterator();i.hasNext();) {
    int integer = ((Integer) i.next()).intValue();
    System.out.println(integer);
}
```

<sup>6</sup> Java 1.2 přinesla několik zásadních změn a novinek v koncepci jazyka; proto se tato i všechny pozdější verze označují jako Java 2

<sup>7</sup> Velmi podrobný přehled změn a novinek lze nalézt také na adrese [java.sun.com/j2se/1.5.0/jcp/beta1/](http://java.sun.com/j2se/1.5.0/jcp/beta1/).

<sup>8</sup> [java.sun.com/j2se/1.5.0/index.jsp](http://java.sun.com/j2se/1.5.0/index.jsp)

Zde programátoři v Javě jen tiše záviděli kolegům používajících C++ jejich šablony. Tento problém mají vyřešit *generické datové typy*. Ty vypadají podobně jako šablony v C++, ale princip je mírně odlišný. Zatímco v C++ se pro každý nový typ vytvoří nová třída, v Javě jde pořád o jednu třídu, ale překladač zajistí typovou kontrolu, zabrání předávání nekompatibilních parametrů a zajistí automatické přetypování návratových hodnot<sup>9</sup>. Příklad pak vypadá takto:

```
List<Integer> list = new ArrayList<Integer>();
// ...
list.add(new Integer(1));
// ...
for (Iterator<Integer> i=list.iterator();i.hasNext();) {
    int integer = i.next().intValue();
    System.out.println(integer);
}
```

### 4.3 Autoboxing a Auto-unboxing

Dalším nepříjemným problémem jazyka Java je existence tzv. *primitivních datových typů* (int, float, boolean, apod.), které jsou nekompatibilní s typy objektovými. To znamená, že je např. není možné vkládat do standardních kolekcí, které umí pracovat pouze s objektovými typy. Tento problém je řešen zavedením objektových variant všech primitivních datových typů (Integer, Float, Boolean, apod.).

Důsledkem tohoto řešení je nutnost explicitní konverze mezi objektovými a primitivními typy; pokud chceme z typu int udělat Integer, použijeme příslušný konstruktor, a pokud chceme získat hodnotu primitivního typu int z instance třídy Integer, použijeme metodu intValue(). Toto je poněkud nepohodlné a může to znepřehledňovat zdrojový kód.

Proto byly zavedeny mechanismy, které se nazývají *Autoboxing* a *Auto-unboxing*, a které tyto konverze provádějí automaticky. Příklad z minulé kapitoly tedy můžeme napsat takto:

```
List<Integer> list = new ArrayList<Integer>();
// ...
list.add(1);
// ...
for (Iterator<Integer> i=list.iterator();i.hasNext();) {
    int integer = i.next();
    System.out.println(integer);
}
```

Pokud by se někdo domníval, že by se to dalo vyřešit i bez Autoboxingu pouhým deklarováním seznamu jako

```
List<int> list = new ArrayList<int>();
```

pak připomínám, že parametrem generické třídy může být opět pouze objektový typ.

<sup>9</sup> Důsledkem tohoto přístupu je to, že je možné použít opět výhradně objektové typy; primitivní typ jako parametr generické třídy užít nelze.

#### 4.4 Rozšířená syntaxe příkazu for

Jak je jistě patrné z předchozích příkladů, velký důraz při přidávání nových konstrukcí do jazyka Java byl kladen na zjednodušení a zpřehlednění zápisu kódu. Dalším logickým krokem na této cestě je zjednodušení způsobu pro procházení (iterování) kolekcemi. Je zavedena nová varianta cyklu for, která nás zbaví nutnosti explicitně užívat iterátor<sup>10</sup>. Náš příklad pak bude vypadat takto:

```
List<Integer> list = new ArrayList<Integer>();
// ...
list.add(1);
// ...
for (Integer i : list) {
    int integer = i;
    System.out.println(integer);
}
```

Je nutné konstatovat, že tento kód vypadá mnohem přehledněji než jeho první varianta a že nové jazykové konstrukce plní svůj účel.

#### 4.5 Výčtový typ

Další věc, která doposud v Javě chybí, je typ *enum*. To se doposud řešilo několika způsoby, od definování konstant typu `int` (což postrádá jakoukoliv typovou kontrolu) až po užívání typově bezpečného řešení založeného na definici třídy pro každý výčtový typ, navrženého v [3]. Toto řešení ve své nejjednodušší variantě vypadá takto:

```
public class Color {
    private Color() {};

    public static final Color red    = new Color();
    public static final Color green  = new Color();
    public static final Color blue   = new Color();
    public static final Color yellow = new Color();
}
```

Právě na tomto principu je založená implementace výčtového typu, která je od verze 1.5 přímou součástí jazyka Java. Výčtový typ se zde ale definuje mnohem snadněji:

```
public enum Color { red, green, blue, yellow };
```

Pro úplnost je nutné poznamenat, že takto definovaný výčtový typ disponuje mnohem širšími možnostmi než předchozí varianta<sup>11</sup>.

<sup>10</sup> Zde je třeba upozornit, že se ve skutečnosti iterátor stále používá; příslušný kód je ale generován překladačem. Všechny uvedené jazykové konstrukce (generické typy, auto-boxing, nová varianta cyklu `for i enum`) jsou řešené výhradně na úrovni překladače; JVM zůstává stále stejná.

<sup>11</sup> viz [java.sun.com/j2se/1.5.0/docs/api/java/lang/Enum.html](http://java.sun.com/j2se/1.5.0/docs/api/java/lang/Enum.html)

#### 4.6 Další vlastnosti

Nových vlastností Javy je celá řada, ale není zde bohužel dostatek prostoru pro každou z nich. Následující vlastnosti patří k těm významnějším, o kterých rozhodně stojí za to se zmínit:

- *Metadata* – tento nový prvek Jazyka Java umožňuje zdrojový kód rozšířit o doplňkové informace deklarativního charakteru; uplatnění nalezne zejména v J2EE, kde eliminuje nutnost vytváření spousty zbytečného kódu a psaní *deployment deskriptorů*.
- *Formátování vstupu a výstupu* – nyní bude v Javě konečně možné používat mocné formátování výstupu funkcí `fprint`, známou z jazyků C/C++. Kromě toho jsou rozšířeny možnosti zpracování vstupu.
- *Proměnný počet parametrů* – toto je další prvek známý z jazyků C/C++; osobně jej považuji za nepříliš důležitý a někteří lidé tvrdí, že byl zaveden jenom kvůli metodě `printf` (viz předchozí bod).
- *Nový Look and Feel* – ve Windows přibyl nový vzhled pro knihovnu swing, který vypadá jako Windows XP, v Linuxu pak vzhled, který vypadá jako GTK.
- *Rozšířená podpora pro XML* – za tímto téměř marketingovým pojmem se skrývá mnoho drobných ale velice užitečných vylepšení. Patří mezi ně plná podpora XML 1.1 a jmenných prostorů, XML Schémat, SAX 2.0.1, XSLT a XSLTC, DOM level 3, XPath, a další.
- *Monitorování a správa* – umožní monitorovat a ovládat virtuální stroj pomocí rozhraní JMX.
- Podpora pro 32-bitové znaky a Unicode 4.0.

### 5 Zajímavé projekty

Projekt *Struts*<sup>12</sup> je rámec pro vývoj WWW aplikací v JSP s využitím Modelu 2 (známý také pod názvem Model View Controller). V současné době je aktuální verze 1.1 a připravuje se verze 2.0. Ta má přinést podporu JSF, což je nový standard JCP pro rámce pro tvorbu uživatelského rozhraní internetových aplikací.

Projekt *Tomcat*<sup>13</sup> (referenční implementace servlet kontejneru z J2EE) poskočil do verze 5.0, která implementuje standardy JavaServlets verze 2.4 a JSP verze 2.0 z J2EE verze 1.4.

*Jboss*<sup>14</sup> je open source implementací J2EE, šířenou pod LGPL licencí. Jde o velmi oblíbený projekt s poměrně intenzivním vývojem. Stabilní verze 3.2 implementuje J2EE verze 1.3, takže je mírně pozadu za J2EE SDK od firmy SUN Microsystems, nicméně vývoj verze 4.0 pokračuje mílovými kroky a čtvrtá generace tohoto aplikačního serveru slibuje mnoho nových lahůdek.

<sup>12</sup> [jakarta.apache.org/struts/](http://jakarta.apache.org/struts/)

<sup>13</sup> [jakarta.apache.org/tomcat/](http://jakarta.apache.org/tomcat/)

<sup>14</sup> [www.jboss.org](http://www.jboss.org)

*JPackage*<sup>15</sup> je další zajímavý a užitečný projekt. Jeho cílem je poskytovat ucelenou řadu balíčků s programy, nástroji a knihovnami v jazyce Java pro nejrozšířenější distribuce Linuxu. Některé balíčky sice nejsou vždy aktuální, ale to je dané konkrétním správcem konkrétního balíku. Trochu mne mrzí, že tento systém balíčků není kompatibilní s oficiálními balíky Javy v mé oblíbené distribuci SuSE.

## 6 Závěr

Jak je nejspíš patrné, i v oblasti platformy Java platí slova klasika: „Jo, jo, pořád se něco děje“. Takže platforma Java rozhodně není na ústupu (jak se ke svému údivu občas dočítám v diskuzních fórech českých internetových magazínů) a vše nasvědčuje tomu, že tu s námi tento perspektivní a užitečný nástroj ještě nějaký pátek zůstane.

Zájemcům o více zajímavých informací o vývoji v této oblasti bych doporučil klasický zdroj informací – Internet. Zejména pak oficiální stránky jazyka Java<sup>16</sup>, stránky TheServerSide<sup>17</sup> nebo Javalobby<sup>18</sup>; z českých zdrojů je asi nejlepší Jablok<sup>19</sup>, dále pak Java.net<sup>20</sup> nebo Dagblog<sup>21</sup>.

## Reference

1. Austin, C.: J2SE 1.5 in a Nutshell.  
<http://java.sun.com/developer/technicalArticles/releases/j2se15/>
2. Branický, M.: Java 2 SE 1.5.0 – technologické preview.  
<http://interval.cz/clanek.asp?article=3215>
3. Bloch, J.: Java efektivně – 57 zásad softwarového experta. Grada, 2002.
4. Pecinovský, R.: Java 1.5: Tygr vystrkuje drápy.  
<http://www.cw.cz/cw.nsf/0/AC950FC7248197DDC1256E9400332C48?OpenDocument&cast=1>

<sup>15</sup> [www.jpackage.org](http://www.jpackage.org)

<sup>16</sup> [java.sun.com](http://java.sun.com)

<sup>17</sup> [www.theserverside.com](http://www.theserverside.com)

<sup>18</sup> [www.javalobby.com](http://www.javalobby.com)

<sup>19</sup> [www.jablok.cz](http://www.jablok.cz)

<sup>20</sup> [www.neo.cz/tomas/java.net/](http://www.neo.cz/tomas/java.net/)

<sup>21</sup> [www.sweb.cz/pichlik/](http://www.sweb.cz/pichlik/)