

Téma 1: Programovací jazyk JAVA

Na počátku zrodu Javy byl vznik **Green Teamu** vedený Jamesem Goslingem, který měl za úkol vytvořit pro společnost Sun Microsystems jazyk, který bude stručný a efektivní. Pro programování systémů domácích spotřebičů.

Green Team se zprvu pokoušel systém pro spotřebiče založit na jazyce C++, poté i na jazyku Pascal. Po neúspěších Green Team navrhl nový jazyk, jenž pochází z C++. Jazyk pojmenovali dle stromu, co rostl před kanceláří Jamese Goslinga **Oak** (Dub). Oak se zprvu nemohl prosadit, po neúspěších v PDA telefonech a set-top boxů k televizorům použít pro webové aplikace – applety. Oak oživil v té době statický web a začal se proto používat na psaní webových appletů.

V roce 1995 se zjistilo, že jazyk se jménem Oak už existuje a tak dostal název **Java**. Java byla poprvé představena v roce 1995 na konferenci SunWord. Technologie Java byla licencována společností IBM, Microsoft, Symantec, Silicon Graphics, Inprise (Borland), Oracle, Toshiba či Novell. V roce 1996 bylo vydáno první Java Development Kit – JDK 1.0, které obsahovalo vše potřebné pro tvorbu appletů.

Java pochází z jazyka C++, byli odstraněni konstrukce, které se nepoužívali, způsobovali těžké běhové chyby a ukazatele do paměti. Někteří programátoři co přešli z jazyka C++ k Javě tvrdí že v Javě jejich produkce programů dvojnásobně stoupá.

Java běží na **Java Virtual Machine**, která zprostředkovává veškeré propojení z hardwarem což zaručuje nezávislost na OS a hardwaru počítače. Ovšem i to má svou nevýhodu, operační systémy musí mít JVM instalovanou popřípadě se musí doinstalovat. Na konci 90. let se totiž Sun dostal do sporu s Microsoftem, který si chtěl Javu upravovat podle sebe. Vše to vyvrcholilo soudním sporem a nakonec odebráním licence Microsoftu. Microsoft přestal do svých operačních systémů dávat aktuální verze JDK a nedával tam žádný. Nyní už Microsoft do svých systémů JDK zabudovává. Teda alespoň já jsem ho tam měl.

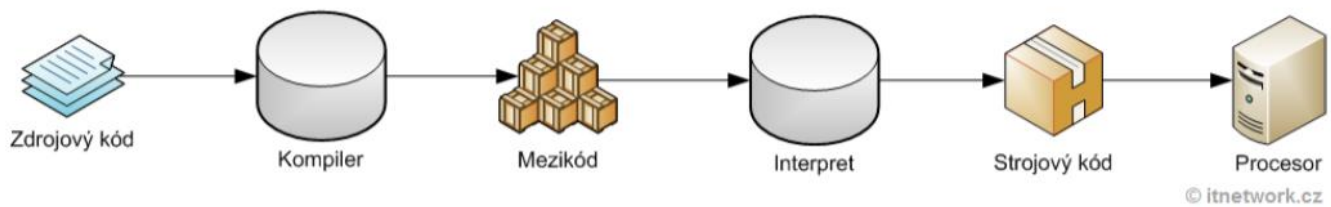
Dnes je Java jeden z nejpoužívanějších programovacích jazyků. Programy v Javě se píšou pro mobily, osobní počítače, servery a internetové aplikace. Největší zastoupení má Java na serverových aplikacích.

Jak již bylo **řeceno** **Java Virtual Machine** obstarává vazbu na hardware a na JVM běží všechny javovské programy. JVM je součástí balíčků jak JDK (Java Development Kit) tak i JRE (Java Runtime Environment).

Java Development Kit obsahuje JVM a také spoustu předepsaných tříd uspořádaných v balíčcích v jedné velké javovské knihovně. JDK je potřeba pro překlad programu.

Java Runtime Environment obsahuje JVM a knihovny stejně jako JDK, hlavní rozdíl je že JRE je určeno jen pro spouštění javovských aplikací, nikoliv pro překlad.

Zdrojový kód je nejprve **přeložen** do tzv. mezikódu, kterému se u Javy říká bytecode. Jedná se v podstatě o strojový (binární) kód, který má ale o poznání jednodušší instrukční sadu a přímo podporuje objektové programování. Tento mezikód je potom díky jednoduchosti relativně rychle interpretovatelný tzv. virtuálním strojem (tedy interpretem, v případě Javy je to tzv. JVM - Java Virtual Machine). Výsledkem je strojový kód pro náš procesor.



Výhody:

- **Odhalení chyb ve zdrojovém kódu** - Díky kompilaci do bytekódu jednoduše odhalíme chyby ve zdrojovém kódu.
- **Stabilita** - Díky tomu, že interpret kódu rozumí, zastaví nás před vykonáním nebezpečné operace a na chybu upozorní. Můžeme také provádět reflexi (i když pro bytekód, ale od toho jsme většinou odstíněni).
- **Jednoduchý vývoj** - Máme k dispozici hitech datové struktury a knihovny, správu paměti za nás provádí garbage collector.
- **Slušná rychlost** - Rychlost se u virtuálního stroje pohybuje mezi interpretem a kompilerem. Virtuální stroj již výsledky své práce po použití nezahazuje, ale dokáže je cachovat, sám se tedy optimalizuje při čtenějších výpočtech a může dosahovat až rychlosti kompilery. Start programu bývá pomalejší, protože stroj překládá společně využívané knihovny.
- **Málo zranitelný kód** - Aplikace se šíří jako zdrojový kód v bytekódu, není tedy úplně jednoduše lidsky čitelná.
- **Přenositelnost** - Asi je jasné, že hotový program poběží na každém železe, na kterém se nachází virtuální stroj.

Další výhodou Javy je, že je zcela zdarma a tedy dostupná všem vývojářům. Aplikace v Javě lze také spouštět přímo ve webovém prohlížeči pomocí Java Web Start. Ten se také automaticky stará o aktualizaci vaší aplikace.

Jazyky s virtuálním strojem ctí objektově orientované programování a jedná se o současný vrchol vývoje v této oblasti. Existují i jazyky 4. a 5. generace, ale ty mají specifické použití.

Zařazení

- **Jednoduchý** – jeho syntaxe je zjednodušenou (a drobně upravenou) verzí syntaxe jazyka C a C++. Odpadla většina konstrukcí, které způsobovaly programátorům problémy a na druhou stranu přibyla řada užitečných rozšíření.
- **Objektově orientovaný** – s výjimkou osmi primitivních datových typů jsou všechny ostatní datové typy objektové.
- **Distribuovaný** – je navržen pro podporu aplikací v síti (podporuje různé úrovně síťového spojení, práce se vzdálenými soubory, umožňuje vytvářet distribuované klientské aplikace a servery).
- **Interpretovaný** – místo skutečného strojového kódu se vytváří pouze tzv. mezikód (bajtkód). Tento formát je nezávislý na architektuře počítače nebo zařízení, který má k dispozici interpret Javy, tzv. virtuální stroj Javy - Java Virtual Machine (JVM). V pozdějších verzích Javy nebyl mezikód přímo interpretován, ale před prvním svým provedením dynamicky zkompileován do strojového kódu daného počítače (tzv. just in

time compilation - JIT). Tato vlastnost zásadním způsobem zrychlila provádění programů v Javě ale výrazně zpomalila start programů. V současnosti se převážně používají technologie zvané HotSpot compiler, které mezikód zpočátku interpretují a na základě statistik získaných z této interpretace později provedou překlad často používaných částí do strojového kódu včetně dalších dynamických optimalizací (jako je např. inlining krátkých metod atp.).

- **Robustní** – je určen pro psaní vysoce spolehlivého softwaru – z tohoto důvodu neumožňuje některé programátorské konstrukce, které bývají častou příčinou chyb (např. správa paměti, příkaz goto, používání ukazatelů). Používá tzv. silnou typovou kontrolu – veškeré používané proměnné musí mít definovaný svůj datový typ.
- **Generační správa paměti** – správa paměti je realizována pomocí automatického Garbage collectoru který automaticky vyhledává již nepoužívané části paměti a uvolňuje je pro další použití. To bylo v prvních verzích opět příčinou pomalejšího běhu programů. V posledních verzích běhových prostředí je díky novým algoritmům pro garbage collection a tzv. generační správě paměti (paměť je rozdělena na více částí, v každé se používá jiný algoritmus pro garbage collection a objekty jsou mezi těmito částmi přesunovány podle délky svého života) tento problém ze značné části eliminován.
- **Bezpečný** – má vlastnosti, které chrání počítač v síťovém prostředí, na kterém je program zpracováván, před nebezpečnými operacemi nebo napadením vlastního operačního systému nepřátelským kódem.
- **Nezávislý na architektuře** – vytvořená aplikace běží na libovolném operačním systému nebo libovolné architektuře. Ke spuštění programu je potřeba pouze to, aby byl na dané platformě instalován správný virtuální stroj. Podle konkrétní platformy se může přizpůsobit vzhled a chování aplikace.
- **Přenositelný** – vedle zmíněné nezávislosti na architektuře je jazyk nezávislý i co se týká vlastností základních datových typů (je například explicitně určena vlastnost a velikost každého z primitivních datových typů). Přenositelností se však myslí pouze přenášení v rámci jedné platformy Javy (např. J2SE). Při přenášení mezi platformami Javy je třeba dát pozor na to, že platforma určená pro jednodušší zařízení nemusí podporovat všechny funkce dostupné na platformě pro složitější zařízení a kromě toho může definovat některé vlastní třídy doplňující nějakou speciální funkčnost nebo nahrazující třídy vyšší platformy, které jsou pro nižší platformu příliš komplikované.
- **Výkonný** – přestože se jedná o jazyk interpretovaný, není ztráta výkonu významná, neboť překladače pracují v režimu „just-in-time“ a do strojového kódu se překládá jen ten kód, který je opravdu zapotřebí.
- **Víceúlohový** – podporuje zpracování vícevláknových aplikací
- **Dynamický** – Java byla navržena pro nasazení ve vyvíjejícím se prostředí. Knihovna může být dynamicky za chodu rozšiřována o nové třídy a funkce, a to jak z externích zdrojů, tak vlastním programem.
- **Elegantní** – velice pěkně se v něm pracuje, je snadno čitelný (např. i pro publikaci algoritmů), přímo vyžaduje ošetření výjimek a typovou kontrolu.