Video tutoriál pro prostředí Eclipse

Eclipse environment videotutorial

Radka Entlová

Bakalářská práce
2011

Univerzita Tomáše Bati ve Zlíně
Fakulta aplikované informatiky

Univerzita Tomáše Bati ve Zlíně
Fakulta aplikované informatiky
akademický rok: 2010/2011

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

(PROJEKTU, UMĚLECKÉHO DÍLA, UMĚLECKÉHO VÝKONU)

Jméno a příjmení:	**Radka ENTLOVÁ**
Osobní číslo:	**A10763**
Studijní program:	**B 3902 Inženýrská informatika**
Studijní obor:	**Informační a řídicí technologie**
Téma práce:	**Video tutoriál pro prostředí Eclipse**

Zásady pro vypracování:

1. **Popište aktuální verzi prostředí Eclipse formou uživatelského manuálu.**
2. **Srovnejte prostředí Eclipse s ostatními vývojovými prostředími.**
3. **Popište základní nastavení prostředí pro práci programátora.**
4. **Vytvořte zásady nastavení prostředí pro práci a tvorbu programů v jazyce Java.**
5. **Vytvořte zásady nastavení pro programování jazyce C++.**
6. **Vytvořte zásady nastavení prostředí pro vývoj webových a síťových aplikací.**
7. **Umístěte vypracovaný tutoriál na DVD jako samostatnou přílohu.**

Rozsah bakalářské práce:

Rozsah příloh:

Forma zpracování bakalářské práce: **tištěná/elektronická**

Seznam odborné literatury:

1. **Domácí stránka programu Eclipse [online, počítačévý program], Ver. programu Eclipse IDE for C/C++ Developers [2010], citováno [20.12.2010], Dostupné z [http://www.eclipse.org/downloads/, http://www.eclipse.org/].**
2. **Domácí stránka programu Netbeans [online, počítačévý program], Ver. programu NetBeans IDE 6.9.1[2011], citováno [5.1.2011], Dostupné z [http://netbeans.org/].**
3. **Domácí stránka programu Visual Studio [online, počítačévý program], Ver. programu Visual Studio 2010 [2011], citováno [5.1.2011], Dostupné z [http://www.microsoft.com/visualstudio/en-us/download].**
4. **Informace a rady o nastavení programu Eclipse [online] [2007-2010], citováno [20.1.2011], Dostupné z [http://www.dagblog.cz/search/label/eclipse].**

Vedoucí bakalářské práce: **Ing. Dalibor Slovák**
Ústav počítačových a komunikačních systémů

Datum zadání bakalářské práce: **25. února 2011**

Termín odevzdání bakalářské práce: **7. června 2011**

Ve Zlíně dne 25. února 2011

prof. Ing. Vladimír Vašek, CSc.
děkan

L.S.

prof. Ing. Vladimír Vašek, CSc.
ředitel ústavu

ABSTRAKT

Cílem této bakalářské práce je vytvořit video tutoriály pro práci ve vývojovém prostředí Eclipse. Práce pojednává o dalších vývojových programovacích prostředích, kterými jsou Microsoft Visual Studio a NetBeans. Popisuje jejich vlastnosti, architekturu, historii, vývoj a jednotlivé edice. Tato práce se zaměřuje na vývojové prostředí Eclipse a popisuje jeho vývojářské prostředí. Uvádí také potřebná nastavení pro práci v programovacích jazycích Java, C/C++ a při vývoji webových aplikací.

Klíčová slova: Eclipse, Microsoft Visual Studio, NetBeans, Java, C/C++, webové aplikace, IDE, vývojové prostředí, tutoriál, video tutoriál

ABSTRACT

The goal of this work is to create video tutorial for guide in Eclipse development environment. The work deals about another development programming environments, Microsoft Visual Studio and NetBeans. This work describes characteristic, architecture, history, development and individual editions. This work puts emphasis on Eclipse and describes its integrated development environment and settings for work in programming languages Java, C / C + + and Web application development.

Keywords: Eclipse, Microsoft Visual Studio, Netbeans, Java, C/C++, web application, IDE, Integrated Development Environment, tutorial, video tutorial

Poděkování

Zde bych ráda poděkovala Ing. Daliboru Slovákovi za pomoc s formálními aspekty řešení a za pomoc s technickými problémy.

Prohlašuji, že

- beru na vědomí, že odevzdáním bakalářské práce souhlasím se zveřejněním své práce podle zákona č. 111/1998 Sb. o vysokých školách a o změně a doplnění dalších zákonů (zákon o vysokých školách), ve znění pozdějších právních předpisů, bez ohledu na výsledek obhajoby;
- beru na vědomí, že bakalářská práce bude uložena v elektronické podobě v univerzitním informačním systému dostupná k prezenčnímu nahlédnutí, že jeden výtisk bakalářské práce bude uložen v příruční knihovně Fakulty aplikované informatiky Univerzity Tomáše Bati ve Zlíně a jeden výtisk bude uložen u vedoucího práce;
- byl/a jsem seznámen/a s tím, že na moji bakalářskou práci se plně vztahuje zákon č. 121/2000 Sb. o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů (autorský zákon) ve znění pozdějších právních předpisů, zejm. § 35 odst. 3;
- beru na vědomí, že podle § 60 odst. 1 autorského zákona má UTB ve Zlíně právo na uzavření licenční smlouvy o užití školního díla v rozsahu § 12 odst. 4 autorského zákona;
- beru na vědomí, že podle § 60 odst. 2 a 3 autorského zákona mohu užít své dílo – bakalářskou práci nebo poskytnout licenci k jejímu využití jen s předchozím písemným souhlasem Univerzity Tomáše Bati ve Zlíně, která je oprávněna v takovém případě ode mne požadovat přiměřený příspěvek na úhradu nákladů, které byly Univerzitou Tomáše Bati ve Zlíně na vytvoření díla vynaloženy (až do jejich skutečné výše);
- beru na vědomí, že pokud bylo k vypracování bakalářské práce využito softwaru poskytnutého Univerzitou Tomáše Bati ve Zlíně nebo jinými subjekty pouze ke studijním a výzkumným účelům (tedy pouze k nekomerčnímu využití), nelze výsledky bakalářské práce využít ke komerčním účelům;
- beru na vědomí, že pokud je výstupem bakalářské práce jakýkoliv softwarový produkt, považují se za součást práce rovněž i zdrojové kódy, popř. soubory, ze kterých se projekt skládá. Neodevzdání této součásti může být důvodem k neobhájení práce.

Prohlašuji,

- že jsem na bakalářské práci pracoval samostatně a použitou literaturu jsem citoval. V případě publikace výsledků budu uveden jako spoluautor.
- že odevzdaná verze bakalářské práce a verze elektronická nahraná do IS/STAG jsou totožné.

Ve Zlíně …….…….

podpis diplomanta

OBSAH

ÚVOD

Tématem mé bakalářské práce je vytvoření video tutoriálu pro prostředí Eclipse. Toto téma vzniklo na základě požadavků vytvořit pro studenty názorný a přehledný manuál. V této práci se snažím názorně popsat základní nastavení pomocí manuálu a několika krátkých videí. Manuál a krátká videa popisují, jak pracovat s vývojovým prostředím Eclipse potřebným pro vytváření projektů a úkolů na Fakultě Aplikované Informatiky na UTB.

Vývojové prostředí, anglicky Integrated Development Environment (IDE), je program, který slouží programátorům ke tvorbě aplikací. Úkolem vývojového prostředí je zvýšit výkonnost programátora, který píše kód, proto je pro tohoto uživatele důležitá určitá znalost tohoto prostředí. Některá prostředí jsou orientována pouze na konkrétní jazyk a zaměřují se na konkrétní vlastnosti daného jazyka. Existují ale prostředí, která jsou určena pro více jazyků, např. Eclipse, NetBeans, Visual Studio aj.

První částí této práce je část teoretická, kde uvádím popis vývojových prostředí. V prvních třech kapitolách se podrobněji zabývám jednotlivými vývojovými prostředími. První z nich je Microsoft Visual Studio, dalším je Netbeans od společnosti Sun Microsystems a Eclipse od společnosti Eclipse Foundation. Práce se snaží názorněji ukázat rozdíl mezi jednotlivými prostředími, proto jsou popsána podrobněji. U jednotlivých vývojových prostředí popisuji základní vlastnosti, architekturu, vývoj, nové i starší verze a různé edice, které jsou dostupné. A ve čtvrté kapitole jsou stručně popsány největší rozdíly mezi zmíněnými prostředími.

Pátá kapitola popisuje jednotlivé základní prvky prostředí Eclipse, V šesté, sedmé a osmé kapitole jsou popsány jednotlivé návody pro vývoj aplikací v Javě, C/C++ a vývoj webových aplikací. Všechny návody jsou doplněny o názorné obrázky vývojového prostředí a pravních oken.

V deváté kapitole je uveden popis program CamStudio, pomocí kterého je vytvořen video tutoriál.

I. TEORETICKÁ ČÁST

1 MICROSOFT VISUAL STUDIO

Microsoft Visual Studio (logo na Obrázku 1.1) [1] je vývojové prostředí od společnosti Microsoft, jehož první verze vyšla v roce 1997. Šlo o spojení několika programovacích nástrojů v jeden celek. Nejnovější verzí je Microsoft Visual Studio 2010. Samotné prostředí lze spustit pod operačním systémem Microsoft Windows, ale jednotlivé aplikace je možné zprovoznit na platformách *Microsoft Windows*, *Windows Mobile*, *Windows CE*, *.NET*, *.NET Compact Framework* a *Microsoft Silverlight*. Ve Visual Studiu můžeme vytvářet konzolové aplikace, aplikace s *GUI* (grafické uživatelské rozhraní), *Windows Forms*[22], webové stránky, webové aplikace a služby.

Obrázek 1.1: Logo Microsoft Visual Studia 2010

1.1 Podporované jazyky Visual Studia

Mezi základní podporované jazyky Visual Studia patří C/C++, Visual Basic .NET a C#. Visual Studio umožňuje podporu i dalších programovacích jazyků prostřednictvím jazykových služeb. Jedná se o balíček *Language Service*, který definuje různá rozhraní. U některých programovacích jazyků, jakými jsou Python, Ruby aj., je potřeba nainstalovat podporu daného jazyka dodatečně.

Visual Studio nabízí uživateli verze jednotlivých prostředí, které jsou zaměřené na určitý jazyk. Jedná se o balíčky Microsoft Visual Basic, Visual J#, Visual C# a Visual C++. Microsoft Visual C++ umožňuje kompilovat kód v jazyce C nebo C++. V této verzi je také podpora strojového i spravovaného kódu a jejich kombinace. Visual C++ podporuje *COM* (*Component Object Model*) a knihovnu *MFC* (*Microsoft Foundation Class Library*). *COM* je technologie pro programování komponent. Visual C++ obsahuje sadu průzkumníků, díky těmto průzkumníkům s pomocí *MFC* může uživatel vytvářet kód a *GUI*. Knihovna *MFC* slouží pro vytváření grafického rozhraní. Vytváří ucelené části

Windows API, čímž zajišťuje kompatibilitu pro většinu operačních systémů Windows. Visual C++ podporuje vestavěné funkce, tyto funkce jsou rozpoznány překladačem, proto je není potřeba implementovat jako knihovny. Visual C++ obsahuje standard pro programování počítačů se sdílenou pamětí.

Microsoft Visual C# nabízí implementaci pro jazyk C#. Tento jazyk byl vyvinut firmou Microsoft, je to vysokoúrovňový objektově orientovaný programovací jazyk. Využívá se při tvorbě databázových a webových aplikací, stránek, webových služeb, formulářových aplikací pro Windows. Visual C# se zaměřuje na *platformu .NET Framework* a na jazykové služby pro podporu jazyka C#.

Implementace Microsoft Visual Basic pro programovací jazyk Visual Basic .NET byla vytvořena v roce 2002. Visual Basic se používá pro vývoj konzolových aplikací a aplikací s grafickým rozhraním pro uživatele. Visual Basic podporuje designér tříd, formulářů nebo dat. Kompilátor VB.NET je součástí *.NET Frameworku*, služby pro vytváření VB.NET projektů jsou pouze součástí Visual Studia.

Existují další produkty Visual Studia. Jedním z nich je Microsoft Visual Web Developer pro tvorbu webových stránek, aplikací pro web a webových služeb. Visual Web Developer používá ASP.NET [23] a podporuje C# a VB.NET. Také umožňuje uživateli vytvořit rozložení grafických komponent pro webové stránky. Dalším produktem je Team Foundation Server, své uplatnění má při týmových projektech. Pracuje jako server a poskytuje verzování, kolekci dat, reporting, záznam projektu a Team Explorer. Team Explorer je klientský nástroj pro služby Team Foundation Server (*TFS*).

Dříve Microsoft vytvářel produkty a projekty, mezi které patří Visual FoxPro, Visual Source Safe, Visual J++ a Visual InterDev. Visual FoxPro je objektově orientovaný a procedurální programovací jazyk, byl vytvořen společností Microsoft a vychází z FoxPro od společnosti *Fox Software*. Používá se pro tvorbu databází, podporuje *SQL* (*Structured Query Language*) dotazy a manipulace s daty. Vývoj tohoto projekt je zastaven od roku 2007, ale zůstane nadále podporován až do roku 2015. Visual SourceSafe je verzovací balíček. Byl zahrnut do dalších produktů společnosti Microsoft a jako samostatný projekt přestal existovat. Microsoft Visual J++ a Microsoft Visual J# jsou implementace pro Javu a překladač Javy pro .NET Framework. Vývoj byl zastaven na základě sporu se společností

Sun Microsystems. Visual InterDev sloužil pro vývoj webových aplikací technologií *ASP* [23] a byl nahrazen Microsoft Visual Web Developer. [2, 3, 4, 5]

1.2 Architektura

Architektura Microsoft Visual Studia umožňuje různá rozšíření. Tato rozšíření zajišťuje *VSPackage*. Každá komponenta, kterou si uživatel přeje přidat, je v balíčku VSpackage, který si uživatel nainstaluje jako službu. Vývojové prostředí poskytuje, zodpovídá a zajišťuje vzájemnou komunikaci *SVSolution, SvSUIShell, SVsShel*l a *UI funkce*. Pro Visual Studio jsou jednotlivé editory, designéry a nástroje vytvořeny jako individuální balíčky VSPackage. Pro přístup k balíčkům se používá COM. *Manager Package Framework* (*MPF*) slouží pro správu obalů rozhraní COM, tyto obaly umožňují psaní balíčku v jazyce .NET. MPF je obsažen v *Software Development Kit* (*SDK*). Všechny tyto služby umožňují tvorbu dalších balíčků pro rozšíření funkčnosti Visual Studia.

Language Service (přel. jazyková služba) zajišťuje podporu programovacím jazykům. Předává podněty VSPackage, které zajistí podporu pro požadovaný programovací jazyk. Takováto podpora může obsahovat zvýraznění syntaxe a párových závorek, doplňování příkazů, typy parametrů a chybové značky pro kompilaci. Tyto vlastnosti nazýváme funkčnost. Aby byla funkčnost dostupná, musí být přidáno rozhraní pro požadovaný jazyk.

Microsoft Source Code Control Interface (*MSSCCI*) slouží k verzování, definuje sadu funkcí pro implementaci funkčnosti jednotlivých verzí. Samotné Visual Studio neobsahuje vestavěnou podporu pro verzování, vychází právě z MSSCCI. V dnešní době existuje několik verzí MSSCCI pro různé verze a edice Visual Studia. [3, 5]

1.3 Nástroje Visual Studia

Visual Studio obsahuje editor kódu, debugger, designer, průzkumník otevřených panelů, editor vlastností, průzkumník objektů, průzkumník řešení, průzkumník týmu, průzkumník dat a serverů. Některé nástroje jsou plovoucí okna, která lze schovat, pokud je zrovna uživatel nepoužívá. Další z nástrojů Visual Studia je tzv. *přírůstková kompilace*, která se realizuje na pozadí, když uživatel píše kód. Visual Studio kód vyhodnocuje a vizuálně informuje uživatele o syntaktických a kompilačních chybách.

Editor kódu lze využít pro všechny podporované programovací jazyky a podporuje zvýraznění syntaxe, *refaktorování*, změny pořadí parametrů, přejmenování proměnných a metod, extrakci rozhraní, automatické dokončování pro proměnné, funkce, metody, cykly a dotazy. Refaktorovaní je proces, kterým se provádějí změny v softwaru, aniž by měli vliv na vnější chování kódu. Snaží se o zlepšení struktury softwaru a zároveň nezavést do systému chyby nové. Tuto podporu zajišťuje *IntelliSense*. Ve Visual Studiu si uživatel může nastavit záložky v editoru kódu pro lepší orientaci. V lepší orientaci a přehlednosti kódu může také uživateli pomoci sbalování jednotlivých bloků kódu, vícepoložková schránka, seznam úkolů, přírůstkové a normální hledání s podporou regulárních výrazů aj. Editor kódu ve Visual Studiu podporuje ukládání šablon opakujícího kódu. Šablony z předchozích projektů mohou být použity v kódu novém.

Debbuger je program, který slouží k nalezení chyb. Debbuger Visual Studia je integrovaný a lze jím debbugovat veškeré programovací jazyky, které Visual Studio podporuje. Funkce *Edit* and *Continue* umožňuje upravovat za běhu debbugování. Debbuger Visual Studia podporuje vícejádrové programy a tvorbu obsahů paměti. Může být přiřazen procesu, který je také sám schopný debbugovat. Debbuger umožňuje zobrazování kódu běžícího procesu, pokud je umožněn přístup k tomuto kódu. Ve Visual Studiu lze použít tzv. *breakpointů* a *watchů*. Breakpoint zastaví probíhající program na zvoleném místě v kódu. Aktivace může být podmíněná nebo nepodmíněná. Watche udává hodnoty proměnných během debbugování.

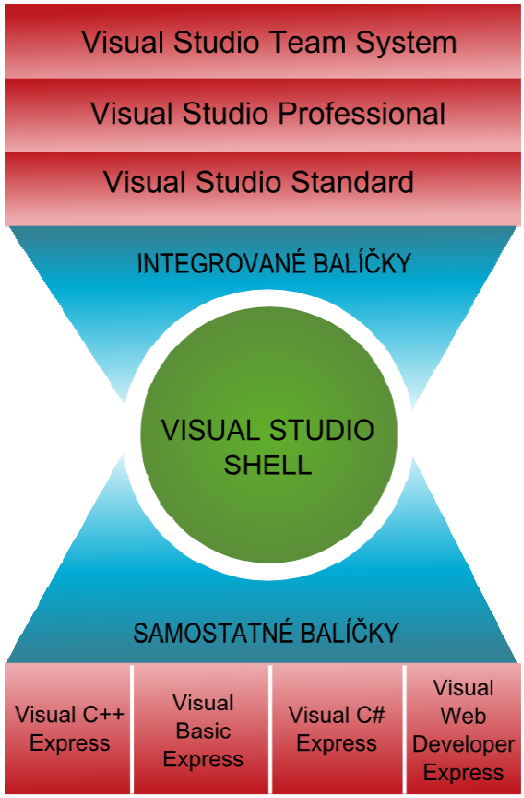
Visual Studio obsahuje různé designery. Těmito designery jsou *WinForms Desinger*, *WPF Designer* [27], *Web designer*, designer tříd, designer dat a designer mapování. Jednotlivé designery slouží uživateli jako usnadnění vývoje aplikací. Pomocí WinForms Designer může uživatel vytvářet GUI aplikací. Obsahuje paletu ovládacích tlačítek, popisků a jiných prvků. Jednotlivé ostatní designery lze využít při vytváření webových aplikací, uživatelských rozhraní, propojení dat, pro vytváření a úpravu tříd modelů *UML*, generování diagramů, úpravu databázových schémat. WinForms Designer také umožňuje mapování mezi databázovými schématy a třídami, které zabalují data.

Dalším nástrojem je průzkumník otevřených panelů, který vypisuje otevřené panely. Průzkumník objektů je dalším nástrojem Visual Studia, jedná se o průzkumníka jmenných prostorů a knihoven tříd pro .NET. Průzkumník řešení ve Visual Studiu umožňuje procházení sad souborů kódu. Průzkumník týmu je integrován s Team Foundation Server,

podporuje kontrolu kódu a možnost spravovat jednotlivé pracovní položky. Dalším průzkumníkem je průzkumník serveru, používá se pro správu připojení k databázi. Editor vlastností umožňuje změnit atributy v GUI panelu. [2, 3, 6]

1.4 Edice Microsoft Visual Studia

Visual Studio existuje ve více edicích, které se liší velikostí, nástroji a odlišnými vlastnostmi. V dnešní době jsou k dispozici edice Visual Studio Express, Visual Studio Standard, Visual Studio Professional a Visual Studio Team System. Visual Studio Express je omezená verze, ve které jsou jazykové služby jednotlivých jazykových verzí nainstalovány na *Visual Studio Shell Appld*, což je jádro vývojového prostředí Visual Studio. Ostatní verze mají jednotlivé jazykové verze integrovány v sobě. Zobrazení vazeb mezi edicemi je na Obrázku 1.2. [2, 3, 7]



Obrázek 1.2: Struktura edic [2]

Visual Studio Express mohou uživatelé používat zdarma, ostatní edice Visual Studia jsou placené. Tato edice je doporučena začínajícím programátorům. Visual Studio Express je sada IDE pro jednotlivé jazyky, jejichž sady jsou samostatně nainstalovány na Visual Studio Shell Appld. Obsahuje pouze omezené nástroje pro konkrétní jazyk,

pro který je určena. Neobsahuje designér tříd, podporu rozšíření, podporu vzdálené databáze aj. Pro verzi Express nejsou k dispozici x64 překladače ani tato verze nepoužívá celou knihovnu *MSDN* (*Microsoft Software Developer Network*). Použitá knihovna vycházející z MSDN knihovny je značně omezená.

Jednotlivé verze jsou:

- **Visual C++ Express**
- **Visual Basic Express**
- **Visual C# Express**
- **Visual Web Developer Express**

Visual Studio Standard Edition obsahuje vývojové prostředí pro všechny podporované jazyky a celou knihovnu MSDN. Nepodporuje vzdálený debbuging, Server Explorer a integraci s Microsoft SQL Server. V této edici lze využít externích nástrojů. V edici Standard může uživatel vytvářet exportovatelné balíčky. Lze zde upravovat *XML* a *XSLT* a objekty *test bench*, což je software sloužící k testování.

Další edicí je Visual Studio Professional Edition, které kromě nástrojů z Visual Studio Standard Edition obsahuje navíc integraci s Microsoft SQL Serverem, vzdálený debugger, neomezené nastavení projektů a Visual Studio Tools for Office. Visual Studio Tools for Office jsou rozšiřující nástroje Visual Studia pro vývoj platformy Microsoft Office.

Visual Studio Team System se dělí podle odlišných sad nástrojů, spolupráce, metrik a vývojového softwaru, který je potřebný pro různé projekty. Obsahuje Team Foundation Server, pomocí kterého lze vzdáleně kontrolovat zdrojový kód, záznam položek kódu a zprávy, složí také ke správě vývojového týmu. Team Suite obsahuje kombinovanou funkčnost těchto různých nástrojů.

Visual Studio Team System se dělí se na:

- **Team Explorer**
- **Architecture Edition**
- **Database Edition**
- **Development Edition**
- **Test Edition**

1.5 Verze Visual Studia

V této kapitole jsou popsány jednotlivé verze, které postupně vyšly. Mezi nejúspěšnější verze patří Visual Studio 6.0, Visual Studio 2003 a Visual Studio 2008. [2, 3, 5]

1.5.1 Visual Studio 97

Visual Studio vyšlo poprvé v roce 1997. První vydání bylo v edici Professional a Enterprise. Poprvé se Microsoft snažil o sloučení jednotlivých programovacích nástrojů pro různé programovací jazyky a také se snažil o jednotné vývojové prostředí pro tyto jazyky. Visual 97 obsahovalo Visual Basic 5.0, Visual C++ 5.0, Visual J++ 1.1 a VisualFoxPro 5.0. Přesto bylo vytvořeno několik vývojových prostředí, které sloužily různým jazykům. Jedno prostředí bylo pro Visual C++ a Visual J++, označovalo se Developer Studio. Oddělená prostředí byla pro Visual Basic a Visual FoxPro. V této verzi byl uveden Visual InterDev, který sloužil k dynamickému generování webových stránek s ASP.

1.5.2 Visual Studio 6.0

V roce 1998 vyšlo na trh Visual Studio 6.0, Microsoft se snažil o lepší rozšiřitelnost. Vývojem prošly všechny části, které přešly na verzi 6.0. Visual Basic, Visual C++ a Visual ForPro měly nově společné vývojové prostředí, které bylo také společné pro Visual J++ a Visual InterDev. Tato verze obsahovala naposledy Visual Basic na základě COM a Visual J++.

1.5.3 Visual Studio .NET 2002

Další verzí po 4 letech bylo Visual Studio .NET. Všechny jazyky byly sjednoceny do stejného vývojového prostředí, které požadovalo platformu na základě Windows NT. V této verzi se poprvé objevilo prostředí pro spravovaný kód s využitím .NET Framework, nový programovací jazyk C# a jazyk Visual J# jako následovník jazyka Visual J++. Nové prostředí mělo lepší rozhraní a jednotnou normu. Používalo okna s nástroji, které se automaticky schovávaly, pokud je uživatel nepoužíval. Zkompilovaný kód se ukládal do formátu *Microsoft Intermediate Language* nebo *Common Intermediate Language*. Aplikace byla poté kompilována při samotném spuštění do strojového kódu pro aktuální platformu, na které běžela, což činilo kód dostupný na více platformách. Tyto platformy ale

musely podporovat Common Language Infrastructure. Kód napsaný v programovacím jazyku J# používá syntaxi jazyka Java, ale může být zkompilován pouze pro .NET Framework.

1.5.4 Visual Studio .NET 2003

Ve Visual Studio .NET 2003 byly částečně odstraněny nedostatky předešlé verze. Verze zahrnovala aktualizovaný .NET Framework. Novinkou byla možnost vytvářet aplikace pro mobilní telefony. Visual Studio .NET 2003 umožnilo vytváření týmových projektů a jejich snazší koordinaci. Tato verze vyšla v edicích Academic, Professional, Enterprise Developer a Enterprise Architect. Enterprise Architect implementoval technologii Microsoft Visio, která vytvářela architekturu aplikace podle UML.

1.5.5 Visual Studio 2005

Visual Studio 2005 nadále zůstalo orientované na aktualizovaný .NET Framework 2.0, přestože .NET z názvu verze i z názvů většiny produktů zmizel. Jednalo se o poslední kompatibilní verzi s Windows 2000. Později vydané aktualizace umožnily kompatibilitu s Windows Vista. Tato verze obsahovala aktualizované a vylepšené prvky předešlé verze, větší podporu *ASP.NET*, lokální webový server, podporu SQL Serveru 2005, Deployment Designera a podporu kompilace pro x86-64 aplikace. Visual Studio Tools for Aplication, jehož předchůdcem byl Visual Basic for Applications, byl vydán pro práci s Office 2007. Je obsaženo jak v Office 2007, tak ve Visual Studiu 2005. Do IntelliSense byly přidány nové typy projektů.

1.5.6 Visual Studio 2008

Hlavním zaměřením Visual Studio 2008 byl vývoj webových aplikací, aplikací pro Windows Vista a Office 2007. Tato verze obsahovala hodně novinek, jakými byly, např. *Windows Presentation Foundation*, nový *HTML/CSS* editor, *XAML* [27] designer, designer pracovního toku, designer *LINQ* [28] do SQL. Při kompilaci si mohl uživatel vybrat, na jaké verzi .NET Frameworku bude aplikace spustitelná.

1.5.7 Visual Studio 2010

Nejnovější verzí je Visual Studio 2010. Podrobnější popis nových vlastností, funkcí a vylepšení je uveden v příloze II. V této verzi je nový editor, který vizualizuje kód. Tato verze obsahuje vestavěné nástroje pro vývoj pro Windows 7, „*drag and drop data binding*“ pro usnadnění tvorby aplikací, také obsahuje různé knihovny pro rychlejší práci. *Parallel Library* slouží pro bezprostřední rozdělení napsaného kódu. *Test Driven Development* (*TDD*) má za úkol usnadnit generování kódu. Ve Visual Studiu 2010 jsou nástroje, pomocí nichž lze vytvářet aplikace pro operační systém Windows Azure (*http://www.microsoft.com/cze/azure/*). [5]

Visual Studio 2010 existuje v následujících edicích [4]:

- **Microsoft Visual Studio 2010 Ultimate** – plná verze, obsahující všechny balíčky a nástroje
- **Microsoft Visual Studio 2010 Premium** – základní edice s rozšířenými nástroji
- **Microsoft Visual Studio 2010 Professional** – základní vývojové prostředí
- **Microsoft Visual Studio 2010 Express** – optimalizované, omezené prostředí
- **Microsoft Visual Studio 2010 Shell** – prostředí pro tvorbu
- **Microsoft Visual Studio Team Foundation Server** – řízení a správa verzí projektu, slouží velkým týmům i jednotlivcům
- **Microsoft Visual Studio Test Professional** – slouží profesionálním testerům

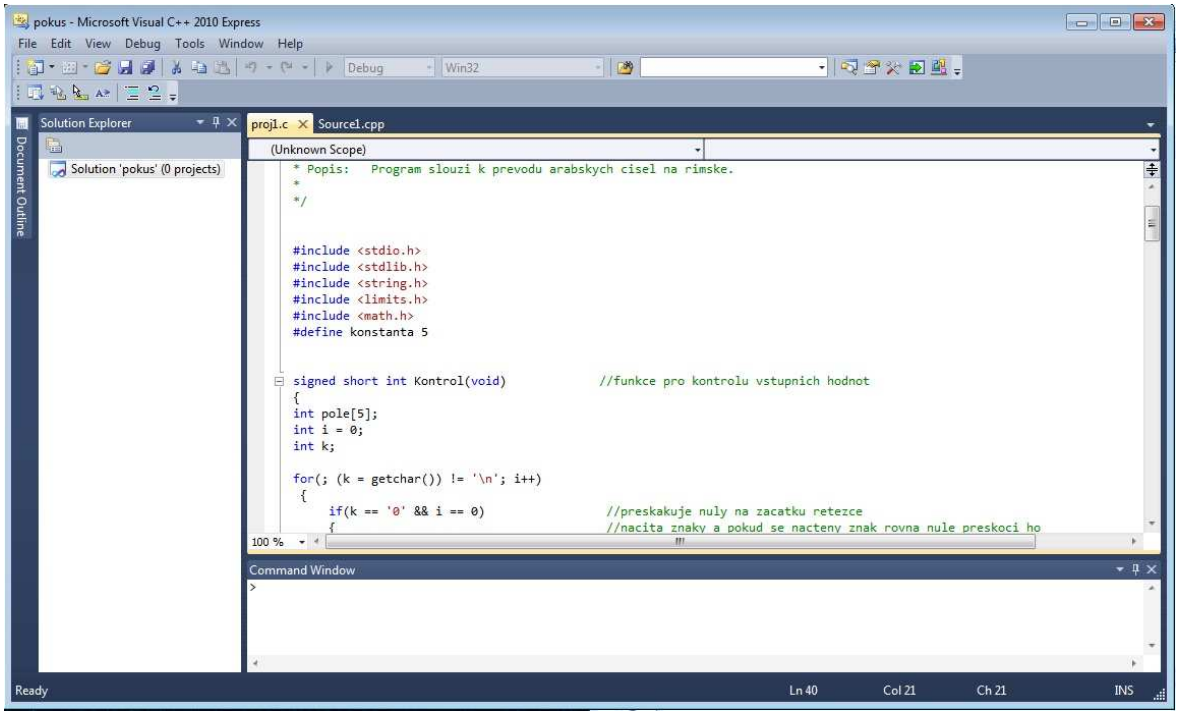
1.6 Práce s Visual Studio 2010

V této podkapitole je popsáno prostředí Visual C++ Express. Na obrázku 1.3 je zobrazeno základní okno Visual C++ Express. Základním prvkem prostředí je *Panel nabídek*. Zde má uživatel k dispozici většinu příkazů, které slouží k ovládání. Pod panelem nabídek je *Panel nástrojů Standard*, který obsahuje tlačítka. Tyto tlačítka jsou zkratky pro správu a ovládání celého prostředí. Standardní příkazy *Otevřít*, *Uložit*, *Vyjmout* a *Vložit* jsou podobné aplikacím Word, Excel apod. Ve spodní části je panel s úlohami systému Windows, lze jej použít při přepínání oken a spouštění jiných programů ve Windows. Je zde, například zkratka pro Internet Explorer, antivir, apod.

Na obrázku 1.3 je zobrazeno úvodní, uvítací okno, kde je pro uživatele uveden stručný návod a několik rad, jak začít. Rovnou zde si uživatel může vytvořit nový projekt nebo otevřít již existující.

Obrázek 1.3: Uvítací okno Visual C++Express

Na obrázku 1.4 jsou zobrazeny okna a nástroje, ale nejsou zde všechny nástroje, které jsou k dispozici ve Visual C++ Express. Ostatní nástroje a okna si může uživatel vybrat v menu. Hlavní částí je *okno návrháře*, kde je na obrázku zobrazen i s již otevřeným projektem a ukázkou kódu. V pravé části obrázku je okno *Solution Explorer*, které lze využít spíše při větších projektech, kdy je potřeba se rychle a přesně orientovat v tomto projektu. Může se zde nacházet *Properties*, *Toolbox*, *Server Explorer*, *Object Browser* aj., záleží na verzi, edici a balíčku Visual Studia. Pokud chce uživatel zobrazit některý z nástrojů nebo oken, může si je zvolit v nabídce *View*, méně časté nástroje se nachází v nabídce *Other Windows*. Velikosti a rozmístění jednotlivých oken a nástrojů si může uživatel jakkoliv přizpůsobit. [8]



Obrázek 1.4: Otevřený projekt ve Visual C++ Express

2 NETBEANS

Historie NetBeans (logo na Obrázku 2.1) [9] sahá do roku 1996, kdy byl vytvořen jako projekt Xelfi na Matematicko-fyzikální fakultě Univerzity Karlovy v Praze. Původně se jednalo o projekt, jehož cílem bylo vytvořit vývojové prostředí pro jazyk Java. Což se také uskutečnilo a Xelfi byl prvním vývojovým prostředím napsaným v Javě a pro Javu. Xelfi se stalo komerčním produktem. V roce 1999 byl již NetBeans odkoupen firmou Sun Microsystems. V červnu následujícího roku zveřejnilo Sun Microsystems NetBeans pod Open-Sources licencí. V roce 2009 byla firma Sun Microsystems prodána firmě Oracle. V současné době se vytváří dva hlavní projekty, a to vývojové prostředí NetBeans (NetBeans IDE) a vývojová *platforma NetBeans* (*The NetBeans Platform*). Oba produkty jsou volně dostupné pod licencí *CDDL* (*Common Development and Distribution License*) [10] nebo *GPLv2* (*GNU General Public License*) [11]. Vývojové prostředí NetBeans je nástroj pro vývoj desktopových, mobilních a webových aplikací. NetBeans je určeno převážně pro programovací jazyk Java, ale podporuje i další programovací jazyky. Vývojové prostředí NetBeans lze bezplatně použít pro nekomerční i komerční účely. Nejnovější verze je NetBeans IDE 7.0, která vyšla v dubnu 2011. Toto prostředí je vytvořeno v jazyce Java a zakládá se na platformě NetBeans. Je spustitelné na platformě *Microsoft Windows*, *GNU/Linux*, *Mac OS X* nebo *Solaris*. [12, 13, 14]

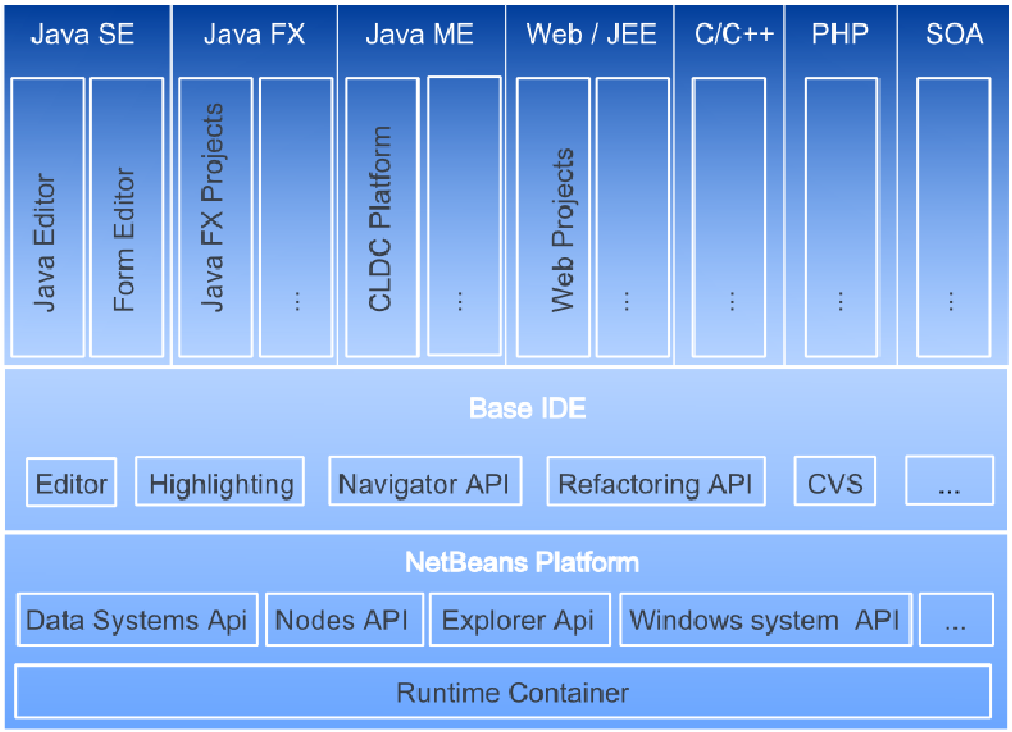
Obrázek 2.1: Logo NetBeans

2.1 Podporované jazyky

Vývojové prostředí NetBeans podporuje hlavně programovací jazyk Java a jeho čtyři hlavní platformy *J2SE* (*Java 2 Standard Edition*), *J2EE* (*Java 2 Enterprise Edition*), *J2ME* (*Java 2 Micro Edition*) a *Java Card* (*Java Card™ 3 Connected*). Dalšími podporovanými jazyky jsou C/C++, PHP. Oficiální podpora jazyka Ruby byla z verze NetBeans IDE 7.0 odstraněna, ale starší verze jazyk Ruby podporují. V NetBeans IDE 7.0 je obsažena podpora pro jazyk Java7 s *JDK* (*Java Development Kit*).

2.2 Architektura

Jádrem vývojového prostředí je *Rich Client* a *NetBeans platforma*. Přehled struktury NetBeans IDE 7.0 je znázorněn na Obrázku 2.2. NetBeans IDE je založeno především na modulech. Rich Client je termín, který se používá v architektuře klient-server pro klienty, kdy se se data zpracovávají na straně klienta. Rich Client může sloužit jako řešení více problémů. Rich Client je postaven na základním rámci (frameworku), který poskytuje určité služby a podporu. To vše slouží uživateli, který sestavuje aplikace pomocí jednotlivých modulů. Moduly jsou logické části, které společně mohou vytvořit velký modul, i když spolu přímo nesouvisejí. Proto mohou výrobci softwaru vytvořit různé aplikace z různých modulů, které splňují specifikace uživatelů. [12, 15]



Obrázek 2.2: Základní struktura NetBeans [15]

Vlastnosti Rich Client aplikace [15]**:**

- *flexibilní a modulární architektura aplikace*
- *nezávislost na platformě*
- *přizpůsobivost*
- *práce na vývoji online nebo offline*

- *jednoduchá aktualizace klienta*
- *snadná distribuce*

Rich Client platforma je software poskytující prostředí pro životní cyklus aplikace. Většina desktopových aplikací má podobné vlastnosti, proto Rich Client obsahuje následující funkce a komponenty: *nabídka, nástrojové lišty, stavový řádek, zobrazení zpracování, zobrazení dat, přizpůsobení nastavení, ukládání dat, načtení dat, úvodní obrazovka, okno O aplikaci, systém nápovědy, aj*.

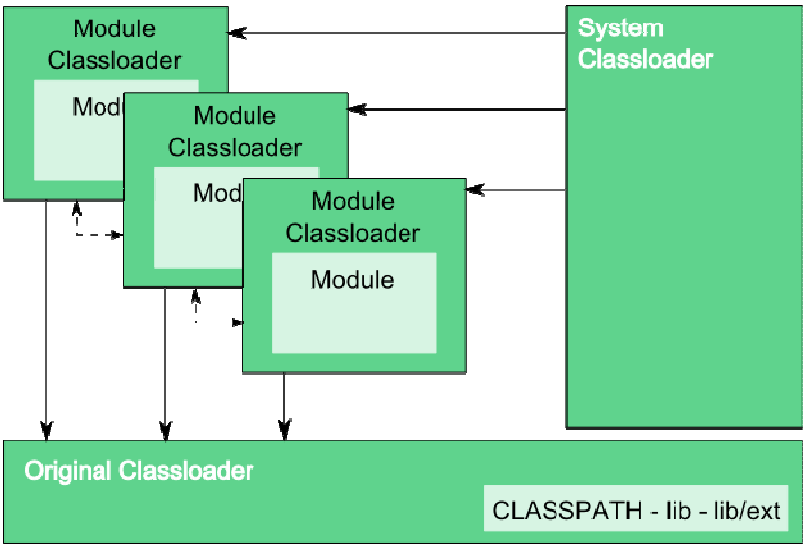
Předností Rich Client platformy je architektura, která umožňuje izolování ucelených částí aplikace, jenž spolu logicky souvisí. Jednotlivé moduly jsou deklarativní, lze je automaticky načíst. To vede k možnosti dynamického rozšíření a snadnějším změnám, např. chování.

Výhody Rich Client platformy [15]**:**

- *Zkrácená doba vývoje* – znovupoužitelné komponenty usnadňující práci
- *Konzistence uživatelského rozhraní* – Rich Client řeší důležité vlastnosti aplikace, kterými jsou konzistentnost, přístupnost a použitelnost
- *Aktualizace* – rychlý a efektivní způsob rozšíření nových modulů a aktualizací
- *Nezávislost na platformě* – dostupné tam, kde běží JRE (Java Runtime Enviroment)
- *Opakované použití a spolehlivost* – většina modulů lze upravit i ve zdrojovém kódu

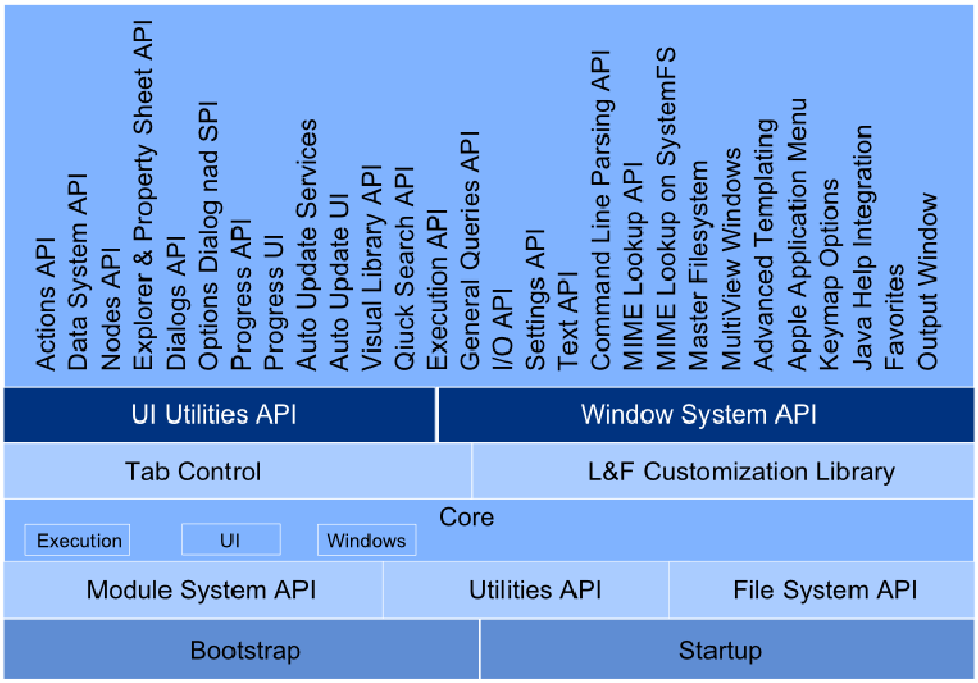
Platforma NetBeans podporuje vývoj a koncepci flexibilních a modulárních aplikací. Jak již bylo zmíněno, platforma NetBeans je založena na modulech. Moduly samotného IDE i projektů jsou sady funkčních svázaných tříd a zdrojů, popisu rozhraní, popis ostatních modulů. Moduly jsou načítány dynamicky a automaticky jádrem platformy, které se nazývá běhový kontejner NetBeans (*NetBeans Runtime Container*). Dovoluje zapouzdření. Jednotlivé části kódu se zapouzdřují v modulech, nutnou optimalizaci zajišťuje samotná platforma NetBeans pomocí vlastních zavaděčů. Načítání modulů zajišťuje zavaděč (*Module Classloader*), jehož činnost je znázorněna na Obrázku 2.3. Dalšími zavaděči jsou systémový zavaděč tříd (*System Classloader*) a prvotní zavaděč tříd

(*Original Classloader*) aj. Modul je načten svým zavaděčem, čímž se vytvoří jednotná ucelená nezávislá jednotka kódu. Tak může modul používat funkce jiných modulů.



Obrázek 2.3: Systém zavaděčů tříd NetBeans [15]

Architektura NetBeans platformy je znázorněna na Obrázku 2.4. IDE obsahuje množství frameworků, API a užitečné specifické vlastnosti. GUI platformy NetBeans je vytvořeno na AWT/Swing a je rozšiřitelné komponentami, které si uživatel může sám vytvořit. Musí deklarovat vzájemnou závislosti, které se zapisují do manifestu modulu. Prostředí NetBeans se stará o to, aby tyto data byla načtena a byla správně použita. [15]



Obrázek 2.4: Architektura NetBeans platformy [15]

2.3 Edice

Vývojové prostředí NetBeans je založené na modulech, proto je možné později přidávat rozšíření, případně podle potřeby i odinstalovat nepotřebné moduly. V následující tabulce (Tabulka 2.1) jsou uvedeny všechny poskytované edice pro NetBeans IDE 7.0. Uživatel si může vybrat z uvedených pěti možností, má buď možnost stáhnout si kompletní vývojové prostředí, které obsahuje vše, nebo i omezené edice pro konkrétní programovací jazyk. [15]

Tabulka 2.1: Přehled jednotlivých edic a jejich podporovaných technologií

Technologie/Programovací jazyky	***Java SE***	***Java EE***	***C/C++***	***PHP***	***Vše***
NetBes Platform SDk	Ano	Ano	Ne	Ne	Ano
Java SE	Ano	Ano	Ne	Ne	Ano
Java EE	Ne	Ano	Ne	Ne	Ano
Java ME	Ne	Ne	Ne	Ne	Ano
Java Card	Ne	Ne	Ne	Ne	Ano
C/C++	Ne	Ne	Ano	Ne	Ano
Groovy	Ne	Ne	Ne	Ne	Ano
PHP	Ne	Ne	Ne	Ano	Ano
Servery					
GlassFish Server Open Source Editon 3.1	Ne	Ano	Ne	Ne	Ano
Apache Tomcat 7.0.11	Ne	Ano	Ne	Ne	Ano

2.4 Historie a verze NetBeans

Od roku 2004 vyšly téměř každý rok dvě verze. V roce 2006 vydalo Sun Microsystems NetBeans 5.0 a na konci roku 2007 NetBeans 6.0. Právě tato verze představila podporu

pro vývoj IDE modulů. Také obsahovala hodně klientských aplikací založených na platformě NetBeans a *Java Swing GUI*. Zlepšila podporu a rozšiřitelnost pro IDE, servery *WebLogic 9* a *JBoss 4*.

NetBeans IDE 6.5 vyšlo v listopadu 2008 a rozšířila stávající funkce Java EE, včetně *Java Persistence*, *EJB 3* (*Enterprise JavaBean*) a *JAX-WS* [24]. Tato verze také podporovala vývoj Java 5, *SOA* (Service Oriented Architecture) vizuální designové nástroje, nástroje pro tvorbu XML schémat, modelování UML aj.

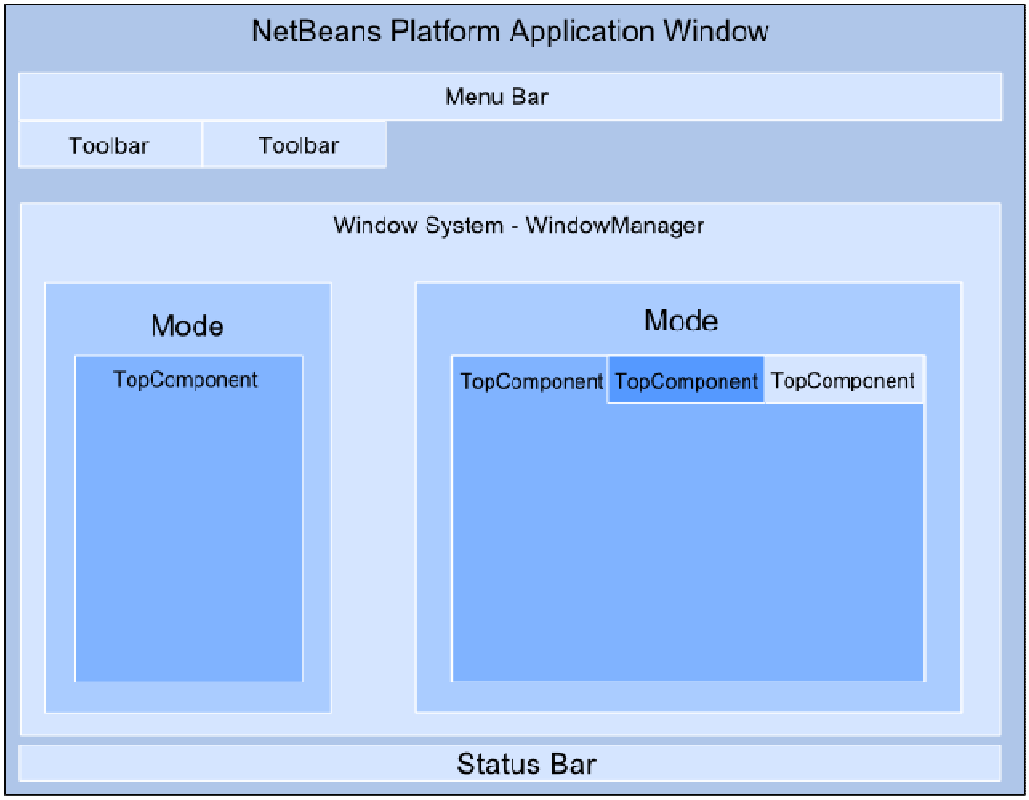
NetBeans IDE 6.8 bylo první verzí, která obsahovala kompletní podporu pro jazyk Java EE 6 a server *GlassFish Enterprise v3*. Pokud vývojář uložil své projekty na kenai.com, mohl využívat výhod komunikačního kanálu, navigace a vyhledávání chyb přímo ve svém vývojovém prostředí. NetBeans IDE 6.8 podporovalo PHP 5.3 a *framework Symfony*. Zlepšilo doplňování kódu, nápovědu a navigaci v projektech *JavaFX*.

V červnu v roce 2010 vyšlo Netbeans IDE 6.9, které obsahovalo navíc podporu *OSGi* (*Open Services Gateway initiative*), *framework Spring 3.0, Framework Zend* pro PHP aj. V této verzi byla vylepšena navigace v kódu, formátování, nápověda aj.

NetBeans IDE 7.0, které vyšlo v dubnu letošního roku, podporuje Java 7, zdokonalený typ interference, jazykovou podporu pro jednotlivé jazykové sbírky, konstanty řetězce v „*case*“, zdokonalený typ inference, *Oracle Database* s jednodušším průvodcem připojení. Také obsahuje lepší integraci s *Oracle WebLogic Server* a *Glassfish 3.1*. Novinkou je také podpora HTML5. V této verzi chybí podpora pro Ruby, na rozdíl od předešlých verzí. V říjnu 2011 je očekáváno vydání verze NetBeans 7.1, které by mělo být synchronizováno s konečnou verzí JDK.

2.5 Nástroje a práce s NetBeans IDE

NetBeans IDE je založeno na okenním systému. Ten se stará o správu nabídek, panelu nástrojů, stavový řádek, grafické rozhraní. Struktura je naznačena na Obrázku 2.5. V horní liště je *Panel Nástrojů* (*Menu Bar*). Pod tímto panelem jsou uživateli nabídnuty zkratky nástrojů (*Toolbar*). Samotná pracovní okna jsou na Obrázku 2.5 značena pod souhrnným označením *Window System – WindowManager*. Ve spodní části je *Stavový řádek* (*Status Bar*).

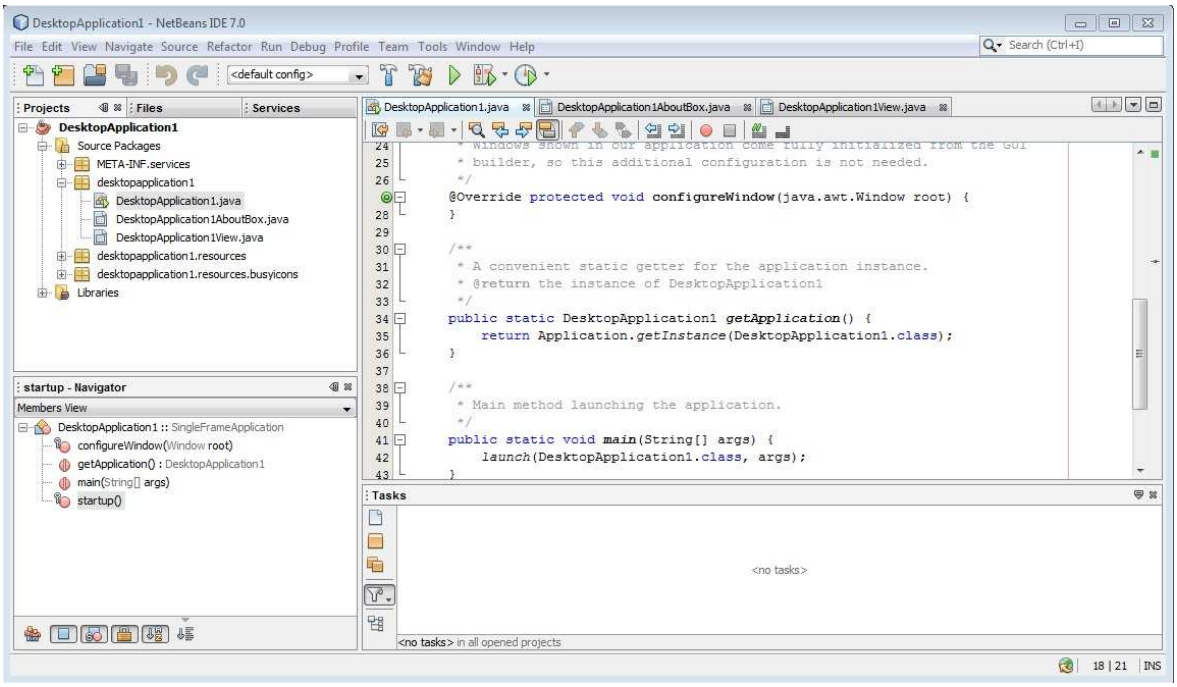


Obrázek 2.5: Struktura aplikačního okna NetBeans IDE [15]

Jednotlivá pracovní okna se mohou lišit od verze i edice, proto se při konkrétnějším popisu prostředí zaměříme pouze na NetBeans IDE 7.0 pro programovací jazyk Java (Obrázek 2.6). Základním nástrojem je editor kódu, kde může uživatel vytvářet zdrojový kód aplikace. Editor kódu obsahuje další nástroje, které uživateli usnadňují samotný vývoj. Základem je zvýrazňování klíčových slov ve zdrojovém kódu. Další funkcí editoru kódu je automatické doplňování názvů metod a polí. Uživatel si tak nemusí pamatovat názvy metod jednotlivých tříd, ale vybere si z nabídky. Ve spojení s programovou dokumentací *Javadoc* nabízí také popis jednotlivých tříd a metod. Další funkcí editoru kódu je automatické formátování zdrojového kódu, který je pak mnohem přehlednější a udržovatelnější. Při psaní kódu aplikace se ihned rozpoznají syntaktické a sémantické chyby a uživatel je na chybu ihned upozorněn barevným bodem na kraji okna editoru.

Dalším oknem ve vývojovém prostředí, které uživatel může využit, je okno Projekty, ve kterém jsou zobrazeny všechny otevřené projekty. Obsah projektu je znázorněn jako stromová struktura, která obsahuje všechny adresáře a soubory v projektu.

Souhrnný přehled atributů a metod každé třídy v projektu je uveden v okně *Navigator*. Zde si uživatel zvolí hledanou metodu či atribut a v editoru kódu se daný prvek zobrazí. V NetBeans IDE existuje mnoho dalších oken jako např. okno Output, ve kterém se zobrazuje výstup právě spuštěné aplikace. Uživatel si může prostředí uzpůsobit podle vlastních potřeb a zvolit si okna, která právě potřebuje v panelu nástrojů v menu Window. NetBeans také obsahuje debugger, kde si uživatel může krokovat průběh běhu programu po jednotlivých řádcích, sledovat hodnoty proměnných a nastavovat breakpointy.



Obrázek 2.6: Ukázka otevřeného projektu v NetBeans IDE 7.0

3 ECLIPSE

Eclipse (Obrázek 3.1) [16] je open source vývojové prostředí, které je vytvářeno neziskovou organizací *Eclipse Foundation*. Toto prostředí se převážně zaměřuje na programovací jazyk Java, ale pomocí pluginů ho lze rozšířit o další podporované programovací jazyky. Nejnovější verzí je Eclipse Helios 3.6.2. Projekt Eclipse vznikal pod vedením firmy IBM, které ho vydalo pod licencí EPL [17] pro volné použití v roce 2001. *Eclipse Foundation* vytváří více jak 60 projektů, které se dělí do sedmi hlavních skupin. První je *Entereprise Development*, který se se zaměřuje na vývoj v J2EE. *Embedded a Device Development* je určen pro vývoj v *J2ME*. Další projekt je pro vývoj na platformě *Rich Client*. *Rich Internet Applications* je určen webovým aplikacím. *Application Framework* umožňuje vytvářet aplikace přímo pro Eclipse. *Application Lifecycle Management* slouží pro práci s daty, např. v databázích. Posledním projektem je *Service Oriented Architecture (SOA)*. Eclipse je k dispozici na operačních systémech *Windows*, *Linux*, *Solaris* aj. [18, 19]

Obrázek 3.1.: Logo Eclipse Helios [16]

3.1 Podporované jazyky a edice

Eclipse je prostředí zaměřené převážně na programovací jazyk Java a dalšími podporovanými jazyky jsou C/C++, PHP, JavaScript, Python. Na stránkách *http://www.eclipse.org/downloads*/ si může uživatel zdarma stáhnout tyto verze a edice Eclipse :

- ***Eclipse IDE for Java Developers*** – základní nástroj pro vývoj aplikace v Javě, obsahuje vývojové prostředí pro Javu, klienta *CVS (Concurrent Version System)*, editor *XML* a *Mylyn* [21]

- ***Eclipse IDE for Java EE Developers*** – nástroj pro vývoj na platformě *Java EE* a pro vývoj webových aplikací, obsahuje vývojové prostředí pro *Javu EE*, *framework JPA* (*Java Persistence API*), technologii *JSF* (*JavaServer Faces*), *Mylyn* aj.
- ***Eclipse Classic 3.6.2*** – obsahuje platformu Eclipse, *JDT* (*Java Development Tools*), pluginy vývojového prostředí, včetně zdrojů a dokumentaci pro uživatele a programátory
- ***Eclipse IDE for C/C++ Developers*** – vývojové prostředí pro vývoj C/C++ s integrací *Mylyn*
- ***Eclipse for PHP Developers*** – nástroj pro vývoj webových aplikací v PHP, obsahuje *PDT* (*PHP Developers Tools*), platformu *Web Tools* [25], *Mylyn* a další
- ***Eclipse IDE for JavaScript Web Developers*** – nástroj pro vývoj webových aplikací pomocí JavaScriptu, obsahuje IDE pro JavaScript, nástroje pro JavaScript, HTML, CSS a XML
- ***Eclipse Modeling Tools*** – balíček, který obsahuje kolekci komponent z projektu Eclipse Modeling Project, obsahuje frameworky *EMF* (*Eclipse Modeling Framework*), *GMF* (*Graphical Modeling Framework*), *M2M* (*Model To Model*), *M2T* (*Model To Text*) a prvky *EMFT* (*Eclipse Modeling Framework Technology*), obsahuje kompletní SDK vývojářské nástroje a zdrojové kódy
- ***Eclipse for RCP and RAP Developers*** – kompletní sada nástrojů pro vývojáře, kteří chtějí vytvářet pluginy, projekty na platformě Rich Client nebo Rich Ajax, obsahuje Mylyn a XML editor
- ***Eclipse IDE for Java and Reportimg Developers*** – nástroj pro Java EE a nástroj zasílání zpráv BIRT pro vývoj v Javě EE a pro vývoj webových aplikací
- ***Eclipse SOA Platform for Java and SOA Developers*** – platforma Eclipse SOA je integrace běhových prostředí a nástrojů pro vývoj na platformě SOA, obsahuje IDE pro Javu, XML editor, *Plugin Development Enviroment* aj.
- ***Pulsar for Mobile Developers*** – nástroj pro vývoj na mobilních platformách, obsahuje plartformu Eclipse, *Java Development Tools* (JDT), mobilní nástroje pro Javu, Mylyn a prostředí pro přidání pluginů, také usnadňuje stažení SDK

3.2 Architektura

Architektura Eclipse je založená na pluginech. Pluginy jsou také nazývány zásuvné moduly, plug-iny aj. Tento software je doplňkovým modulem jiného softwaru a samostatně nepracuje. Rozšiřuje funkčnost, podporu apod. V eclipse jsou pluginy základním stavebním kamenem a nejmenší jednotkou, kterou si uživatel může sám nainstalovat. Každý plugin definuje tzv. extension points, které mají za úkol vymezit možnosti rozšířené jednotlivých vlastností. Eclipse se skládá z platformy Rich Client. Samotná architektura se skládá z hlavní platformy, *frameworku OSGi* (*Open Services Gateway initiative*), *SWT* (*Standard Widget Toolkit*), *JFace* a *Workbench*, které si dále trochu přiblížíme.

Pro platforma Rich Client (RCP) je důležitá hlavní aplikační třída, která implementuje rozhraní *IApplication*. Přepokládá se definování této třídy pomocí *extension point org.eclipse.core.runtime.application*. Další důležitou součástí je perspektiva. Perspektiva je kontejner, jenž soustřeďuje různé editory, průzkumníky a okna. Tyto jednotlivé editory a okna lze libovolně uspořádat a přepínat mezi nimi podle potřeb a přání uživatele. Rozmístění těchto komponent je po spuštění stejné. Průzkumníci a okna, které jsou zařazeny v rámci perspektivy, slouží jako navigace, zobrazení informací nebo pomocí nich lze otevřít jednotlivé editory. Editory slouží k úpravám kódu. Neviditelnou komponentou je *Workbench Advisor*, který dohlíží na viditelnost ostatních komponent, kterými jsou menu, nástrojové panely aj. Podrobněji popsaná architektura Rich Client je v kapitole 2.2 této práce.

Framework OSGi je specifikací dynamického modulárního systému pro programovací jazyk Java, který se považuje za nejvyspělejší modulární systém pro programovací jazyk Java. Framework OSGi umožňuje práci s moduly během vykonávání programu, poskytuje zázemí pro spolupráci jednotlivých modulů, určuje životní cyklus modulu. Skládá se z několika vrstev.

SWT je knihovna pro Javu, která byla vyvinuta IBM a nyní je vyvíjena organizaci Eclipse Foundation. Obsahuje grafické prvky, které mohou uživatelé volně využívat ve svých projektech. JFace je nástroj pro tvorbu uživatelského rozhraní, které obsahuje třídy pro obsluhu mnohých běžných problému při programování uživatelského rozhraní. Podrobnější informace o JFace naleznete na [26].

3.3 Historie a verze Eclipse

Eclipse začalo jako projekt firmy IBM v Kanadě. Původně se jednalo o vývojové prostředí pro jazyk Smalltalk. V roce 2001 se změnila licence na open source a teprve v lednu 2004 vznikla organizace Eclipse Foundation. V červnu 2004 také vyšla verze Eclipse 3.0, která už byla založena na frameworku OSGi. Eclipse 3.1 vyšlo o rok později. Každý rok v červnu je vydána nová verze. Eclipse Callisto 3.2 bylo vydáno 30. června 2006, následovalo Eclipse Europa 3.3 v roce 2007. V červnu 2008 spatřila svět verze Eclipse Ganymede 3.4 a v roce 2009 Eclipse Galileo 3.5. Aktuální verzí je Eclipse Helios 3.6, která vyšla v roce 2010. Nová verze bude vydána v červnu 2011.

Eclipse bývalo dlouho průměrným vývojovým prostředím, které se ale každý rok zlepšuje. Podrobný popis jednotlivých verzí je na stránkách Eclipse *http://www.eclipse.org /documentation/*.

Dokumentace pro jednotlivé verze:

- **Eclipse 3.6 Helios** - *http://help.eclipse.org/helios/index.jsp*
- **Eclipse 3.5 Galileo** - *http://help.eclipse.org/galileo/index.jsp*
- **Eclipse 3.4 Ganymede** - *http://help.eclipse.org/ganymede/index.jsp*
- **Eclipse 3.3 Europa** - *http://help.eclipse.org/help33/index.jsp*

II. PRAKTICKÁ ČÁST

4 SROVNÁNÍ RŮZNÝCH VÝVOJOVÝCH PROSTŘEDÍ

V této práci jsou podrobněji poukázány rozdíly jednotlivých vývojových prostředí Eclipse, NetBeans a Microsoft Visual Studio, proto je každému z nich věnována samostatná kapitola.

Hlavní rozdíl mezi Visual Studiem a zbylými dvěma prostředími je zaměření na programovací jazyk. Visual Studio se převážně zaměřuje na programovací jazyk C/C++, na rozdíl od NetBeans a Eclipse, které se zaměřují hlavně na vývoj v Javě, kterou Visual Studio nepodporuje. Visual Studio není v oblasti Javy konkurence schopným prostředím. Microsoft se proto snaží vyvíjet programovací jazyk J#. Všechna tato prostředí samozřejmě poskytují široký rozsah nástrojů pro mnoho programovacích jazyků. Dalším podstatným rozdílem je, že Visual Studio se všemi užitečnými nástroji je placené. Sice poskytuje verze a edice, které jsou zdarma, ale ty jsou do značné míry omezené.

V dnešní době se zaměřuje vývoj prostředí na pluginy, které poskytují externí dodavatelé. Toto nejvíce podporuje Eclipse Foundation , a proto také Eclipse obsahuje mnoho moderních a žádaných technologií. Zaměření na pluginy občas ztěžuje uživatelům ovládání, ale přináší lepší podporu pro ostatní programovací jazyky.

NetBeans i Eclipse obsahují bohatou dokumentaci a podporu nejen pro vývoj v Javě, i když je pro Javu samozřejmě lépe a podrobněji propracované. Eclipse poskytuje jednotlivá rozšíření pomocí pluginů, NetBeans také nabízí pluginy, ale je jich mnohem méně než u Eclipse. Vývojové prostředí NetBeans je spíše intuitivní, jednoduché a snaží se dodržet konvence IDE oproti Eclipse, které je spíše nekonvenční svými perspektivami.

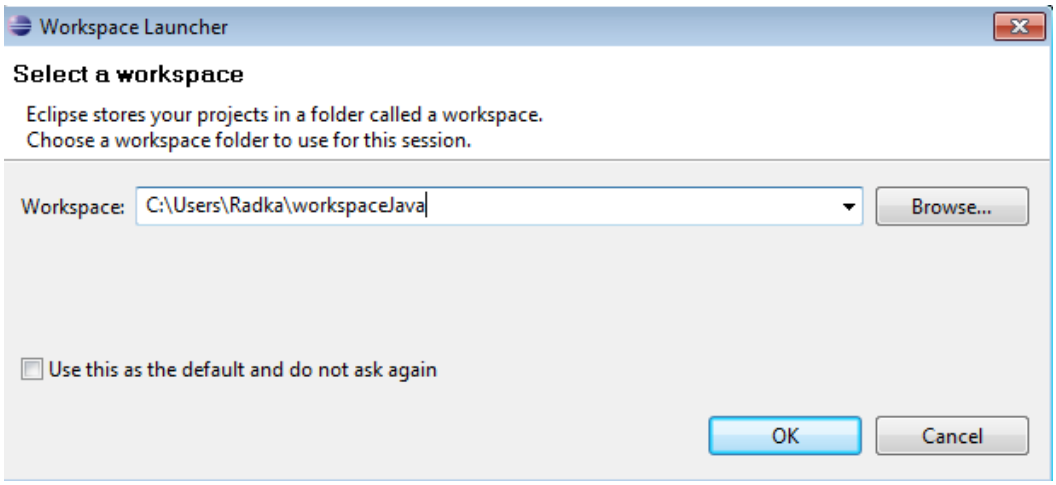
Podle průzkumů IDE Javy, které jsou každý rok zveřejňovány v BZ Research, je Eclipse nejpoužívanější IDE pro Javu, NetBeans je hned na druhém místě. [15, 30]

5 POPIS PROSTŘEDÍ ECLIPSE

V této kapitole je popsáno samotné prostředí Eclipse 3.6. Helios a jeho základní nástroje. Bude zde uvedeno, jak postupovat od počátku nastavení až po práci s Eclipse. Je nutné si uvědomit, že Eclipse bude vypadat na různých operačních systémech odlišně, protože Eclipse je závislé na platformě. V tomto tutoriálu je Eclipse spouštěno na operačním systému Windows 7. [32, 33]

5.1 První spuštění

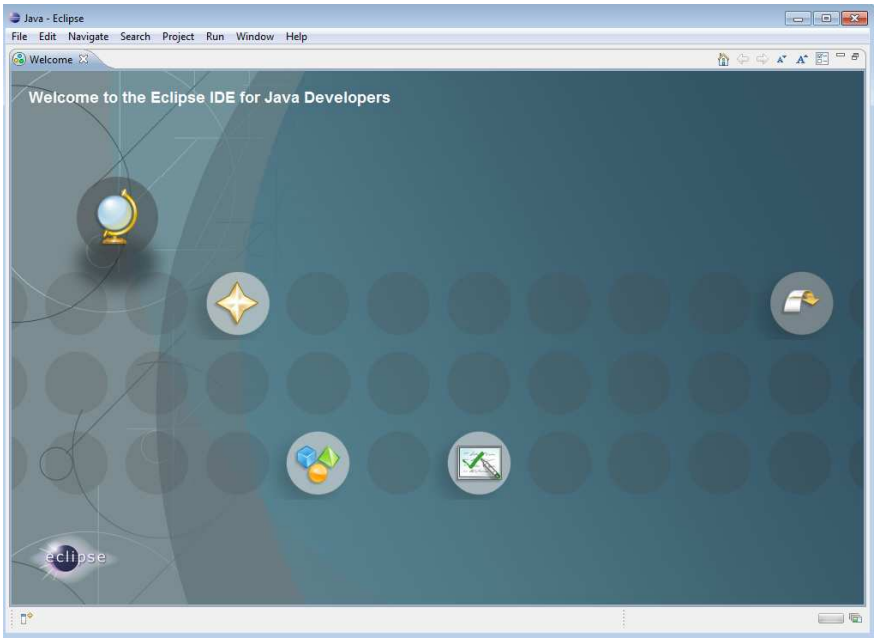
Pro spuštění je potřeba spustit Eclipse pomocí souboru s příponou *.exe*. Než se spustí samotné prostředí Eclipse, musí se zadat cesta k Workspace (Obrázek 5.1). Jedná se o místo, kam se budou ukládat vytvořené projekty a soubory. Pokud se bude využívat vždy stejné místo, musí se zaškrtnout možnost „*Use this as the default and do not ask again*“. Po odsouhlasení se objeví uvítací obrazovka Eclipse, která nabízí pět možností (Obrázek 5.2):



Obrázek 5.1: Nastavení Workspace

- **Overview** – jedná se o ikonu, která ukáže uživateli nápovědu, ve které jsou mimo jiné také základní informace o Eclipse
- **What's New** – tato ikona má informativní charakter, uvádí novinky, které byly uvedeny od aktuální spuštěné verze
- **Samples** – uvádí ukázky kódu a práce s Eclipse

- **Tutorial** – ikona, která uživateli ukáže praktické použití, ukazuje základy i pokročilou práci s Eclipse
- **Workbench** – tato možnost otvírá pracovní prostředí, pokud uživatel zavře toto okno křížkem v pravém horním rohu, dosáhne stejného efektu



Obrázek 5.2: Uvítací obrazovka Eclipse

5.2 Popis pracovního prostředí Eclipse

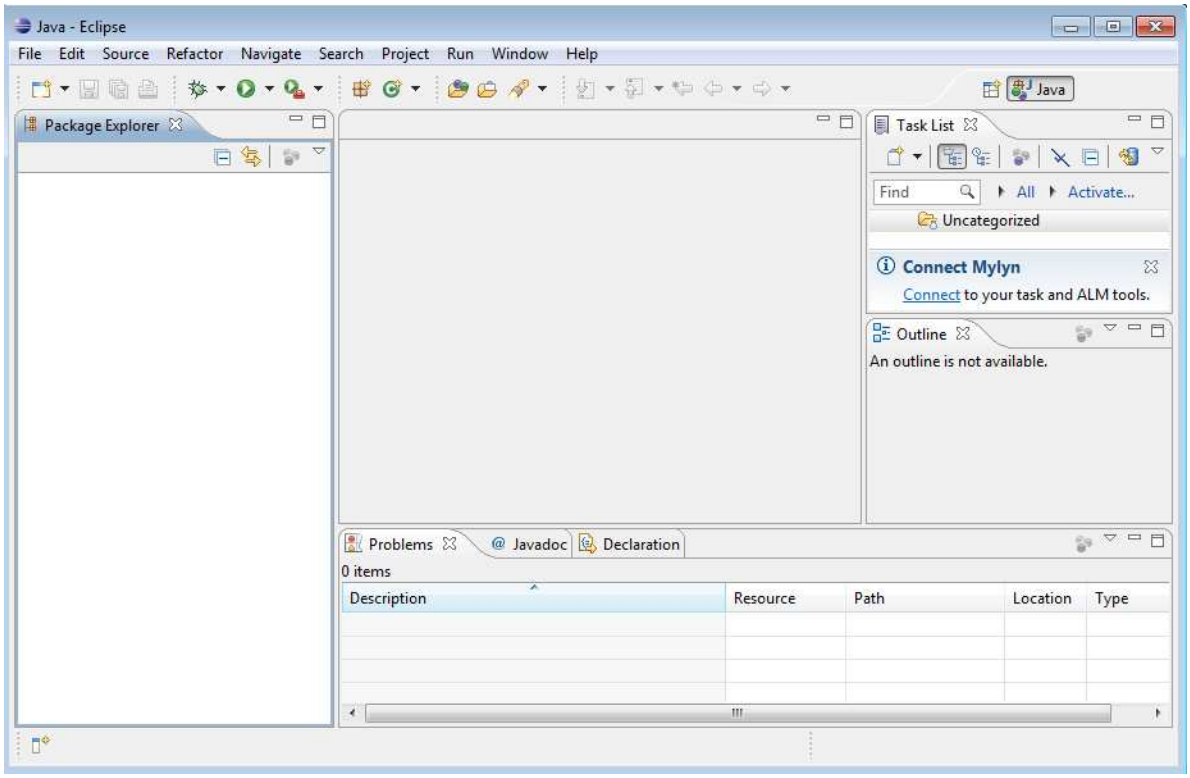
Po uvítací obrazovce se otevře základní pracovní prostředí Eclipse (Obrázek 5.3). Toto pracovní prostředí se dělí do několika základních částí, které jsou uvedeny v následujících podkapitolách.

5.2.1 Menubar (Hlavní nabídka)

Menubar nebo také *hlavní nabídka* je panel, který obsahuje nástroje, funkce a ovládání celého programu rozčleněné do jednotlivých položek a podmenu.

5.2.2 Toolbar (Hlavní nástrojová lišta)

Toolbar neboli *hlavní nástrojová lišta* se nachází pod hlavní nabídkou a obsahuje pouze některé nástroje a ovládací prvky, které by měl uživatel při vytváření projektů používat nejčastěji. Tato lišta je nastavitelná a uživatel si ji může upravit.



Obrázek 5.3: Pracovní prostředí Eclipse

5.2.3 Perspektivy

Jak už bylo zmíněno, základem zobrazení v Eclipse jsou perspektivy. Je možné je popsat jako okna nebo prvky obrazovky, kterými jsou pohledy (*view*) a editory (*editors*). Při práci se přechází z jednoho okna do druhého podle potřeby, aktivní okno se pozná podle výraznější barvy. Použité nástroje se většinou vztahují k aktivnímu oknu. Akce *Vyjmout* (*Cut*), *Kopírovat* (*Copy*) a *vložit* (*Paste*) lze vyvolat stisknutím pravého tlačítka myši na zvoleném pohledu.

Pohledy podávají náhled na určitou část problému, mohou to být data, kód, struktura, vztahy a vazby, aj. Podávají uživateli určitý druh prezentace aktuálního stavu srozumitelnou formou. Základní představitel těchto pohledů je *Package Explorer* (*Průzkumník*), který podává uživateli informace o jednotlivých vytvořených projektech. Každý pohled má tři typy menu. První menu, které slouží k nastavení vlastností a úpravě okna, se vyvolá pravým tlačítkem myši na horní liště okna. Druhé menu je rozbalovací a zobrazí se levým tlačítkem myši po kliknutí na trojúhelníček v pravé části, obsahuje různé nástroje, kterými lze upravit obsah. Třetí typ se vztahuje k jednotlivým prvkům vnitřního

okna průzkumníku. Jeho obsah se vztahuje vždy k danému prvku, který si uživatel zvolil pro vyvolání tohoto menu.

Druhou skupinou jsou editory, které slouží k úpravě souborů. Pro různé edice a verze existují odlišné editory, které mají zvláštní nástroje a funkce pro konkrétní programovací jazyk. Editory se vybírají automaticky podle typu souboru, pokud takovýto editor není obsažen nebo soubor není podporován, otevře se tento soubor v obyčejném textovém editoru. Při prvním spuštění je editor umístěn ve středu hlavního okna, obsahuje záložku se jménem otevřeného souboru. Symbol „*“ u názvu upozorňuje na změnu tohoto souboru, signalizuje, že provedené změny nebyly uloženy. S otevřeným editorem jsou v lištách aktivní pouze nástroje, které lze použít pro daný typ souboru, ostatní jsou nedostupné. Uživatel může editor otevřít několika způsoby. Jedním z nich je otevření souboru, spuštěním položky *New Editor* v podmenu Window z nabídky hlavního menu. Další možností je dvojklik levého tlačítka na soubor v otevřeném projektu v okně Průzkumníku. Uživatel může mít otevřeno více různých editorů.

5.2.4 Package Explorer (Průzkumník)

Jak už bylo dříve zmíněno, *Package Explorer* nebo také *průzkumník* slouží k pohledu, zobrazení struktury a správě projektů, knihoven a souborů. Uživatel si může podrobněji procházet obsah jednotlivých projektů, které lze označit jako otevřené nebo uzavřené, které se značí symboly „-“ a „+“. Otevření a uzavření projektu lze provést kliknutím myši, nebo pomocí položky Open Project nebo Close Project v menu.

5.2.5 Hierarchy (Hierarchie)

*Hierarch*y, česky *hierarchie*, se nachází ve stejné oblasti jako *Průzkumník* a lze ji vyvolat přepnutím záložky. Zobrazuje hierarchii projektů, balíků, zdrojových adresářů apod. Ukazuje náhledy nadřazených typů a podtypů. Hierarchii, např. souboru, lze zobrazit přetažením požadovaného souboru do okna *Hierarchy*.

5.2.6 Outline

Outline se vztahuje k otevřenému souboru v editoru. Jedná se o další zobrazovací okno, které ukazuje podrobnější pohled, udává i vnitřní strukturu s proměnnými apod. Slouží pro programování hlavně v programovacím jazyce Java.

5.2.7 Spodní panel

Spodní panel obsahuje různé záložky různých oken, editorů a nástrojů. V základním zobrazení jsou pohledy *Problems*, *Javadoc* a *Declaration*. Pokud uživatel spustí přeložený kód, objeví se ve spodním panelu pohled *Console*. Tento pohled představuje výstup přeloženého a spuštěného kódu. V pohledu *Problems* jsou uživateli zobrazeny případná varování a problémy vztažené k editovanému souboru. V pohledu *Problems* jsou vypsány typy chyb nebo upozornění a po kliknutí na danou událost se tato událost zobrazí v editoru kódu. Pohled *Javadoc* je program pro psaní programátorské dokumentace tříd, kterou vytváří v HTML kódu. Pohled *Declaration* udává seznam a informace prvků deklarovaných v kódu.

5.2.8 Editor kódu

Editor kódu patří do střední perspektivy, jak již bylo uvedeno dříve. Otevře se sám po otevření souboru a zobrazí kód. Hlavní úlohou editoru kódu je psaní a modifikace kódu v souboru. Editor nabízí zvýraznění a barevné odlišení syntaxe, což napomáhá uživateli v lepší orientaci. V dnešní době editory nabízejí i jiná usnadnění práce, např. automatické doplňování klíčových slov a automatické formátování. Blíže bude editor kódu popsán v následujících kapitolách.

5.2.9 Vlastní nastavení prostředí

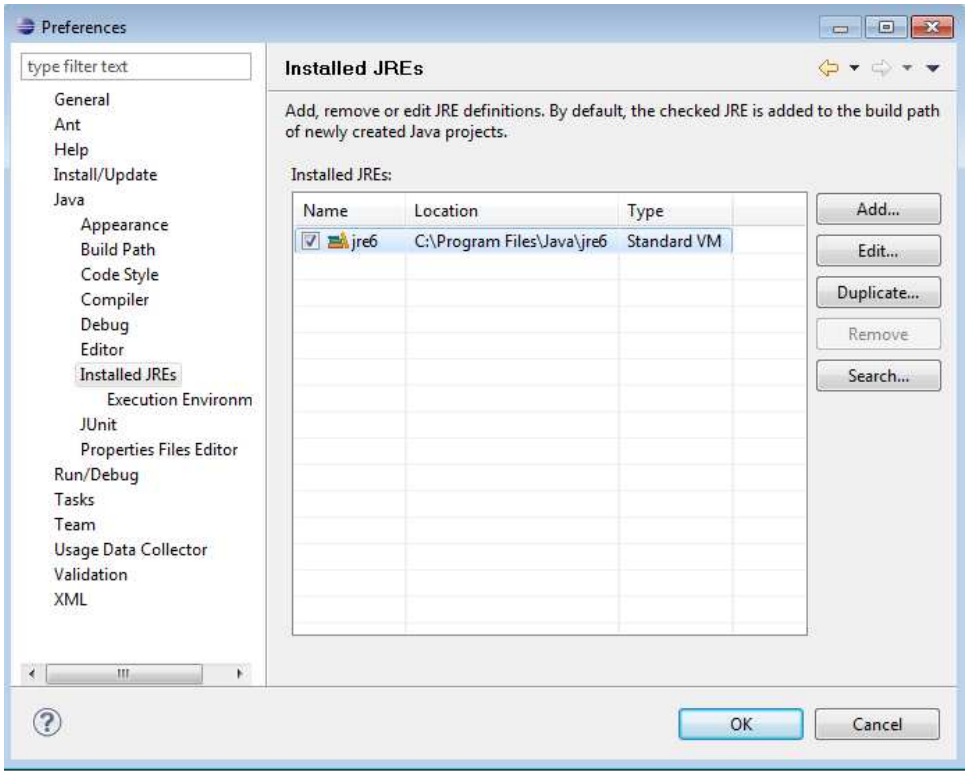
Uživatel si může přizpůsobit vývojové prostředí Eclipse podle svých potřeb. Vlastní nastavení si uživatel zvolí pomocí hlavní nabídky v nabídce *Window*, v tomto podmenu se musí vybrat možnost Show *View*, kde se zobrazí základní nabídka pohledů, pokud tam hledaný pohled není, je potřeba vybrat možnost *Other*. Pro zobrazení konkrétního pohledu stačí kliknout levým tlačítkem na požadovaný pohled. Další možností, jak si upravit vzhled je možnost *Customize Perspective*, která zobrazí okno s dalšími nastavitelnými vlastnostmi. Lze zde předdefinovat funkce a nástroje pro lištu nástrojů a hlavní menu, které se budou automaticky zobrazovat po spuštění Eclipse. *Reset Perspective* nastaví původní zobrazení a prvotní nastavení vývojového prostředí. Nastavení lze uložit mezi již existující perspektivy a lze jej použít pro budoucí nastavení ostatních perspektiv. Pomocí *Navigation* se může uživatel orientovat mezi perspektivami.

6 NASTAVENÍ PROSTŘEDÍ PRO PRÁCI V JAVĚ

Pro vývoj aplikace v programovacím jazyce Java se používá Eclipse IDE for Java Developers. Na této stránce je nutno vybrat požadovanou verzi, edici a platformu, na které se Eclipse spustí. Stažený soubor je typu ZIP a po dekomprimaci lze Eclipse spustit. Tento tutoriál je vypracován v prostředí IDE for Java Developers na platformě Windows 7. Pro práci s programovacím jazykem Java je potřeba předem nainstalovat *Java Development Kit* (JDK). JDK je sada nástrojů, které slouží pro vývoj aplikace, spouštění a kompilaci. JDK lze stáhnout z [29]. [32, 33]

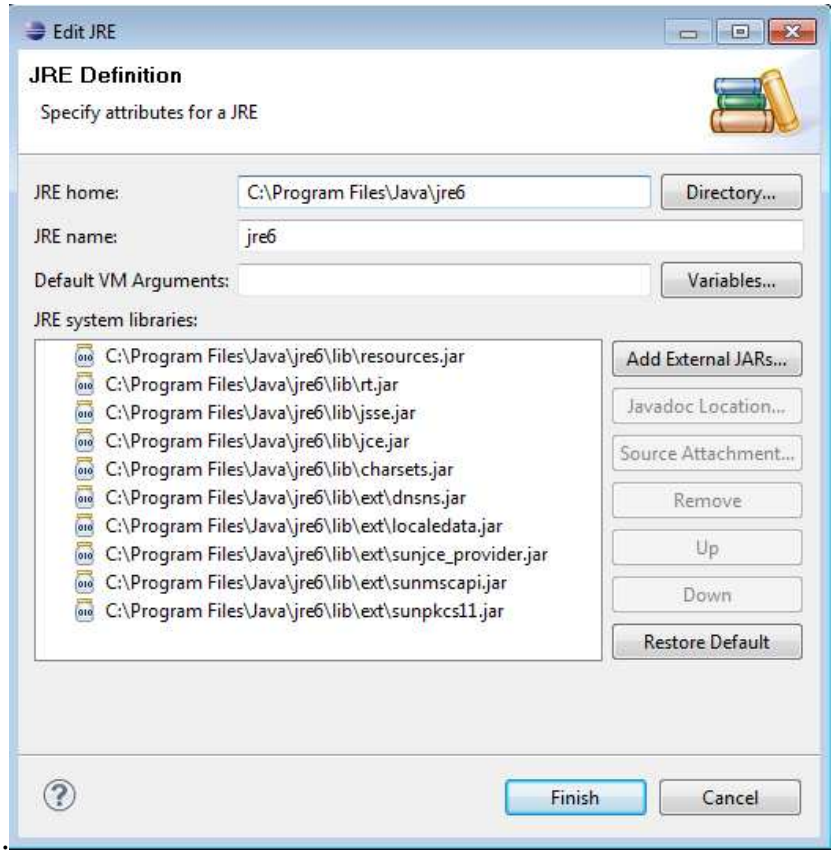
6.1 Počáteční nastavení pro programovací jazyk Java

Nejdříve je nutné nastavit prostředí pro běh Javy, vybere se v hlavním menu položka *Window* a poté se zvolí možnost *Preferences*. Zobrazí se okno *Preferences* (Obrázek 6.1), ve kterém je potřeba zvolit možnost *Java > Installed JREs*. Poté se zaškrtne možnost *JDK* (aktuální nainstalovaná verze) a stiskne se *Edit.. .* V novém okně *Edit JRE* (Obrázek 6.2) se musí nastavit v poli JRE *name* jméno, poté se vybere možnost *Add External JARs…*, která otevře další okno s výběrem složky uložení JDK.



Obrázek 6.1: Okno Preferences

Poté, co se vše potvrdí, se zobrazí původní okno *Edit JRE* s již aktualizovanými úpravami. I toto okno je třeba potvrdit, čímž se provede konečné nastavení a Eclipse bude využívat JDK.



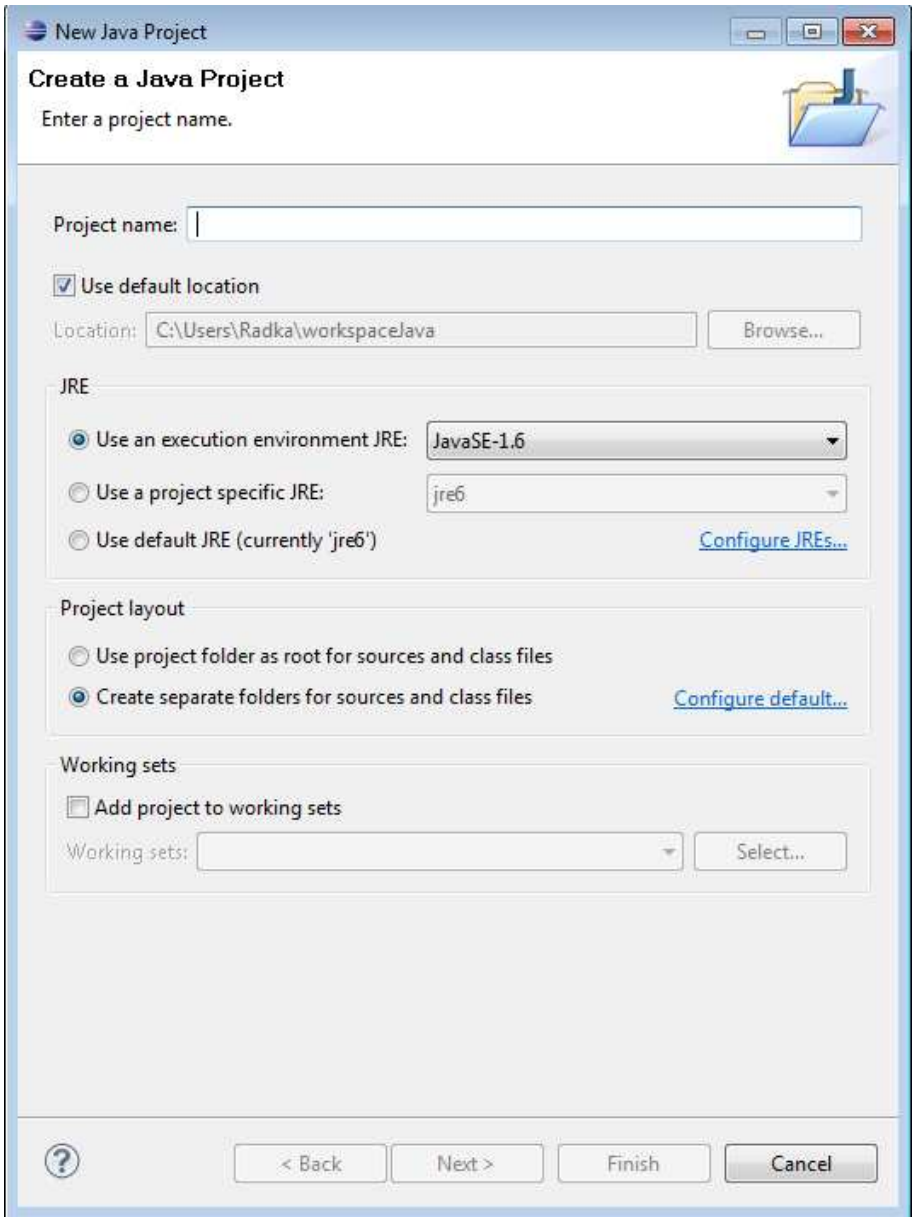
Obrázek 6.2: Okno Edit JRE

6.2 Vytvoření projektu

Vytvoření projektu v Eclipse lze udělat několika způsoby. V hlavní nabídce spuštěním *File > New*, kde se vybere možnost *Java Project*. Další možnost je v *Průzkumníku* pravým tlačítkem myši vyvolat menu, ve kterém se zvolí možnost *New*. I v tomto případě se zvolí možnost *Java Project*. Poslední možností, která je zde uvedena, je využití nástrojové lišty a ikonky , opět se vybere možnost *Java Project*.

Samotný projekt se ale ještě nevytvoří, nejdříve se objeví okno (Obrázek 6.3), ve kterém si uživatel nastaví vlastnosti Java projektu. Nesmí se zapomenout uvést jméno projektu v kolonce *Project name*. V políčku *Contents* se vybírá uložení projektu, volba *Create new project in workspace* uloží nový projekt do složky, která se nastavila při prvním spuštěním programu. Druhá volba *Create project from existing source* otevře

průzkumníka pro výběr již existujícího zdroje. V poli *JRE* se v předešlé kapitole nastavilo JDK, proto se tato možnost ponechá. Pomocí *Configure default* lze opět nastavit JRE nebo novější verzi JDK. Možnost *Use a project specific JRE* dovoluje nastavit projektu jiné JRE, než které je implicitně nastaveno. *Project layout* nastavuje zdrojový a cílový adresář vytvářenému projektu. Zdrojový adresář je to místo, kam se budou ukládat soubory, cílový adresář určuje místo uložení zkompilovaného projektu a jeho částí. Pro první možnost *Use project folder as root for sources and class files* bude zdrojový i cílový adresář ten stejný. Druhá možnost *Create separate folders for sources and class files* nastaví dva různé adresáře.

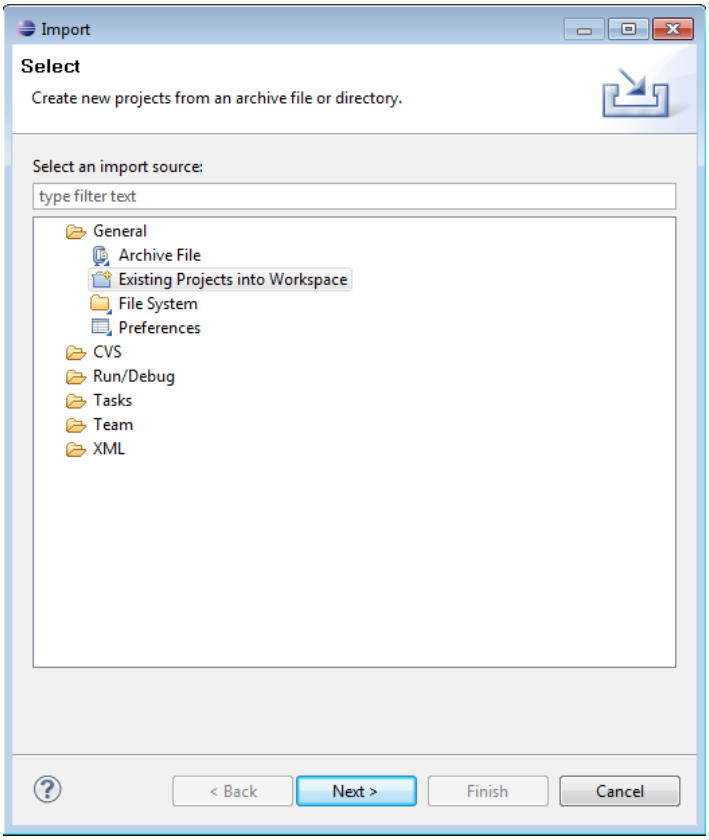


Obrázek 6.3: Nastavení nového projektu

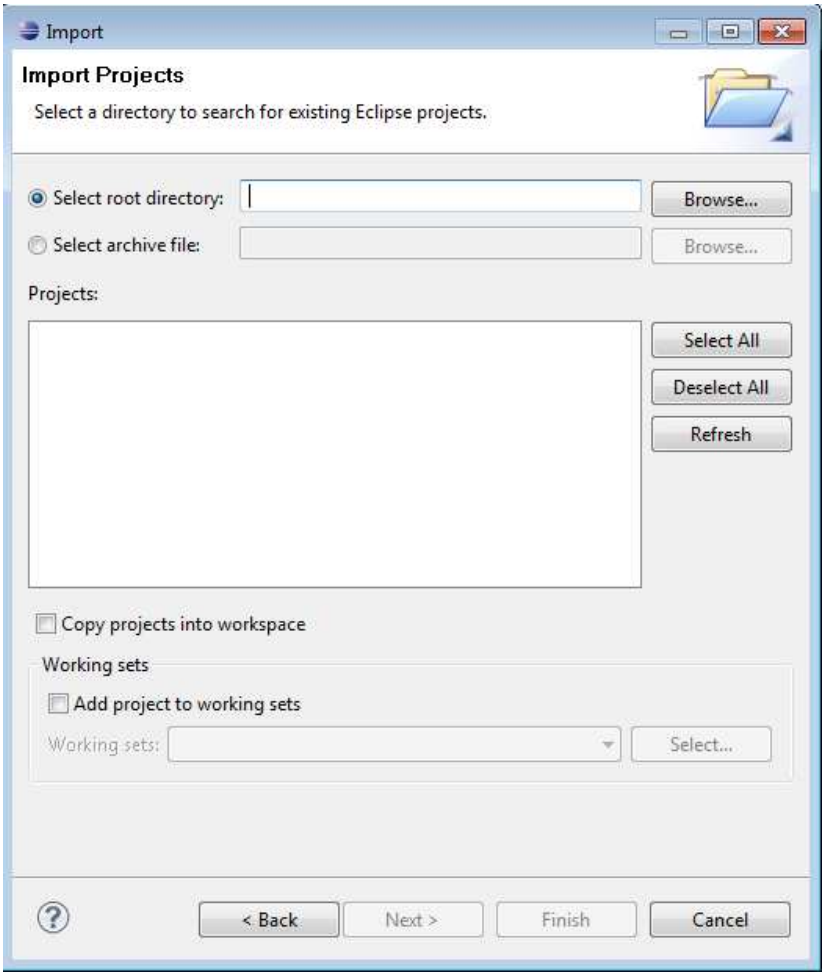
V poli *Working sets* je pouze možnost *Add project to working sets*, čímž se aktivuje pole *Working sets*, kde jsou uživateli nabídnuty pracovní sady pro zařazení projektu. Ve většině případů, kdy bude potřeba vytvořit nový projekt, stačí zadat pouze jméno souboru a ostatní ponechat beze změny. Projekt se standardním nastavením se vytvoří po kliknutí na tlačítko *Finish*. Je tu ale i jiná možnost a tu nabízí tlačítko *Next*, po jehož stisknutí se otevře okno s dalším nastavením projektu. Po úspěšném vytvoření projektu se v *Průzkumníku* objeví položka, která znázorňuje projekt.

6.3 Import existujícího projektu

Importování projektu se provádí několika způsoby. Jedním je výběr možnosti v menu *File*. Druhou možností je vyvoláním menu pomocí pravého tlačítka myši. Při tomto způsobu importování je nutné vybrat možnost, že se jedná o existující projekt. Další možností je vytvořit nový projekt, ale při vytváření zvolit *Create project from existing source*. Poté je jen potřeba doplnit cestu k projektu a název projektu. Dále po dokončení vytvoření projektu se objeví okno *Import*, kde lze zadat cestu k projektu. Po kliknutí na *Finish*, se otevře importovaný projekt v *Průzkumníku*. Projekty lze přidat i z archívu.



Obrázek 6.4: Okno Import



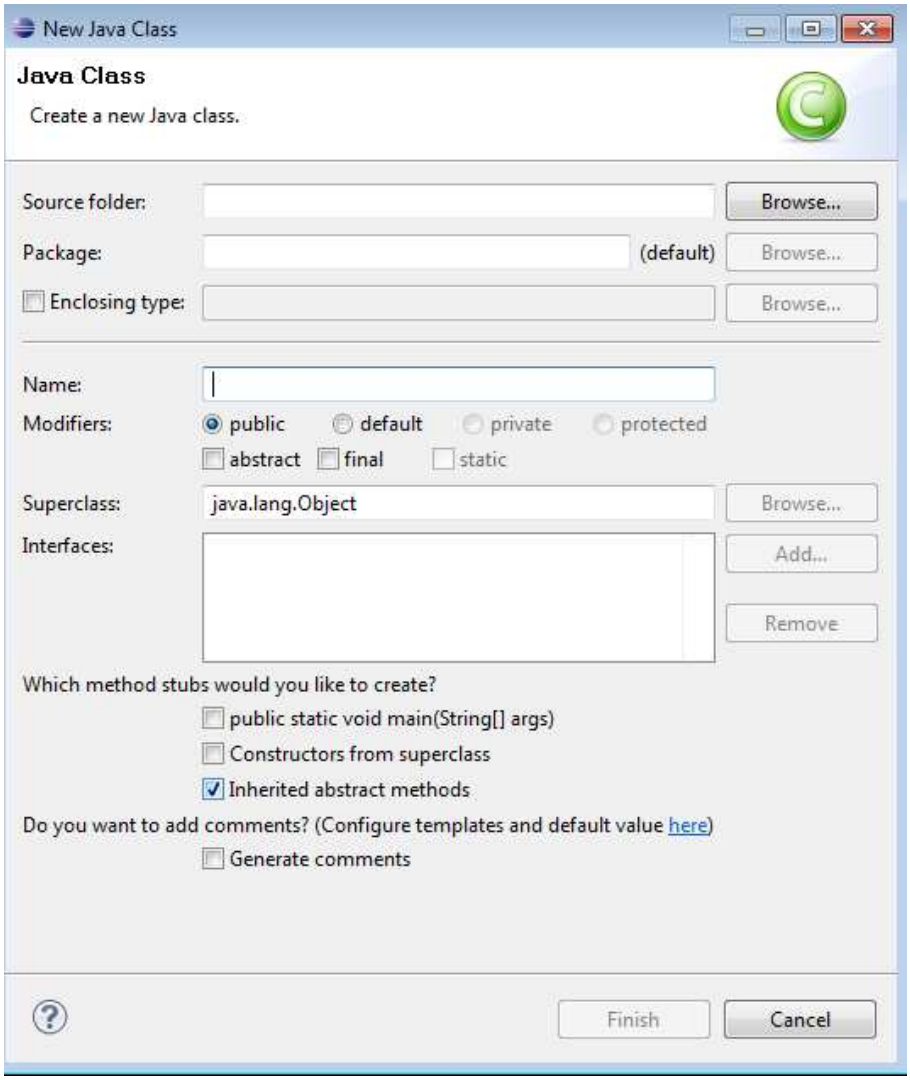
Obrázek 6.5: Okno pro import projektu

6.4 Vytvoření nového souboru

Po vytvoření projektu je důležité vytvořit soubor. Projekt může obsahovat hodně souborů, které lze rozdělit do balíků. Vytvořit soubor je možné opět několika způsoby. Za prvé lze vytvořit soubor pomocí menu *File*, kde se vybere možnost *New* a *Class*. Také lze opět vyvolat menu *Průzkumníka* pravý tlačítkem a opět vybrat možnost *New* a *Class*. Další možností je kliknout na ikonu v hlavním panelu nástrojů, otevře se rozbalovací menu, ve kterém se vybere opět *Class*.

Po provedení jakéhokoliv předešlého způsobu se otevře okno (Obrázek 6.6) pro nastavení vlastností nového souboru. V poli *Source folder* je uveden zdrojový adresář, který patří do projektu a lze jej změnit, ať už přepsáním nebo pomocí *Browse…* . Do pole *Package* se zapíše jméno balíku, do kterého soubor bude patřit. Toto pole ovšem není nutné vyplnit, a potom se nastaví umístění do standardního balíčku. Dalším polem je

Name, kde se zadá název souboru, ale je nutné mít na paměti dodržování konvencí, protože Eclipse si vytvoří již třídu se stejným jménem.



Obrázek 6.6: Vytvoření nového souboru

Pole *Modifiers* složí k nastavení modifikátorů třídy, které slouží při programování v Javě. V poli *Superclass* lze uvést předka vytvářené třídy a v poli *Interfaces* se doplňují rozhraní implementované třídou, které se přidají pomocí *Add…* .

Dále se zadává typ metody, která se má vygenerovat. Lze zde zvolit automatické doplnění typů metod, první metodou *public static void main (String [] args)* se vygeneruje metoda *main*. Druhá metoda Constructors from superclass vytvoří konstruktory předka. Třetí možnost *Inharited abstract methods* vygeneruje abstraktní metody z třídy předků.

Zaškrtnuté pole *Generate commments* doplní základy komentáře. Pro vytvoření jednoduchého souboru není nutné vše vyplňovat. Pro vytvoření souboru je potřeba

dokončit tyto kroky tlačítkem *Finish*, poté se soubor objeví v baličku, který patří do projektu.

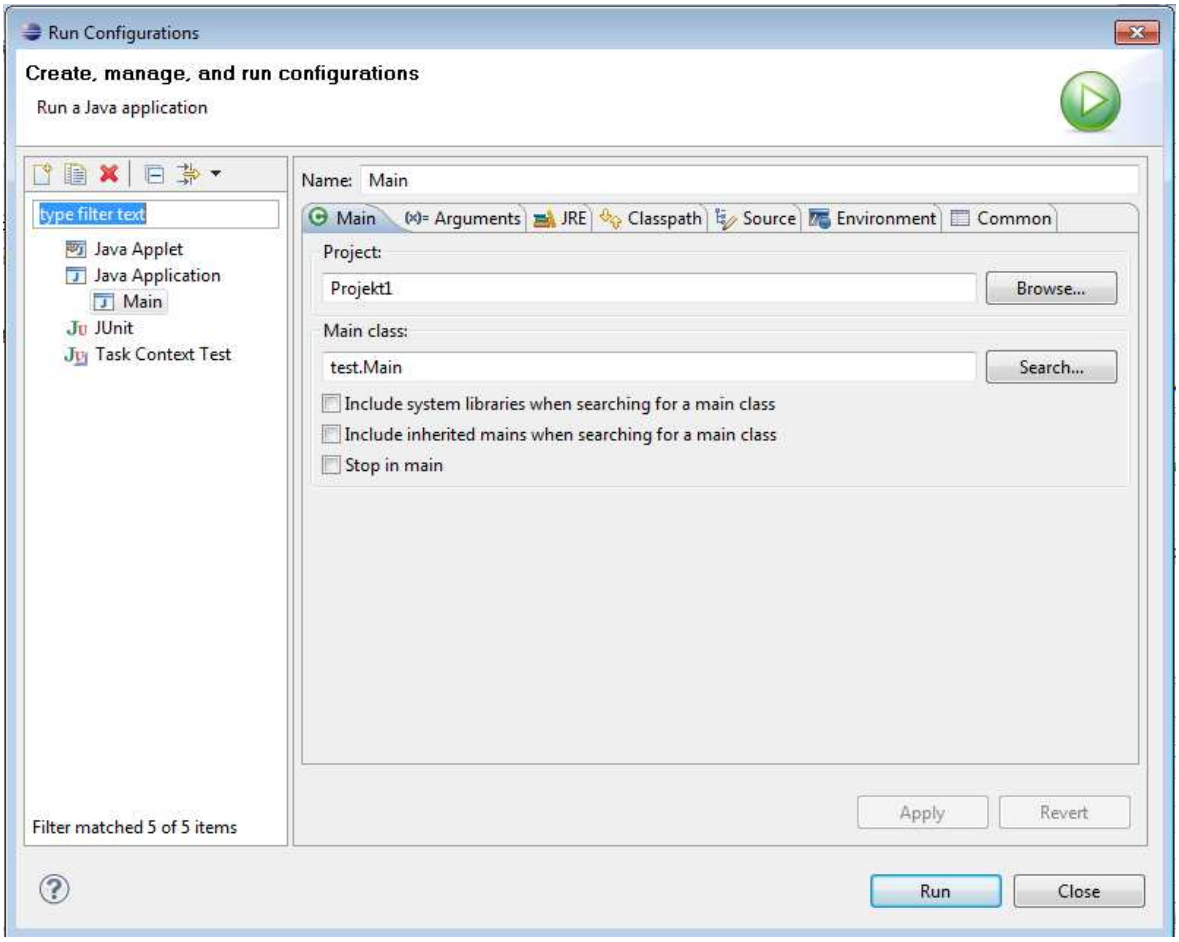
6.5 Vložení existujícího souboru do projektu

Vložení do projektu již existujícího souboru lze pět provést několika způsoby. První způsob je trochu nešikovný, provede se pomocí Windowsovského nástroje Průzkumník, kde se nalezne složka s projektem a nový soubor se jednoduše přesune do této složky. U druhé metody stačí zkopírovat soubor ze složky pomocí *Ctrl + C* a v Eclipse v Průzkumníku stačí do označeného balíku zvolit klávesovou zkratku *Ctrl + V*, nebo pravým tlačítkem myši kliknout na vybraný balík a dát možnost *Paste (Vložit)*. Vložení souboru lze i přetažením myši ze složky přímo do *Průzkumníku* v Eclipse a požadovaného balíčku.

6.6 Kompilace a spuštění programu

Před kompilací je nutné všechny změny v projektu uložit, jinak Eclipse nedovolí kompilaci a bude se dožadovat uložení změn, jinak projekt zkompiluje bez nových změn. Program lze zkompilovat pomocí *Run* v menu *Run*, nebo kliknutím na ikonku *Run* v nástrojové liště. Pomocí nabídky *Run As* je nutné vybrat možnost *Java Application*. Další volbou z menu je *Run History*, které nabídne seznam projektů, které byly spuštěny už předtím. Další z možností menu *Run* je *Open Run Dialog…* (Obrázek 6.7), kde si uživatel může podrobněji nastavit vlastnosti spuštění programu. V levé části tohoto okna je vidět stromová struktura, jejíž podstrom *Java Application* obsahuje všechny spustitelné aplikace. Pole *Name* obsahuje název kompilace. Níže je několik záložek a první z nich je *Main*, která obsahuje pole *Project* a *Main Class*. Pole *Project* obsahuje název kompilovaného projektu, v *Main Class* se vybírá spustitelná třída. Eclipse využívá vlastnosti dané třídy, proto by se hodnoty měli nastavit automaticky. V ostatním záložkách není nutné momentálně nic nastavovat pro tento jednoduchý projekt.

Eclipse spouští vždy třídu, která je aktivně označena v Průzkumníku nebo aktivní třídu v editoru kódu. Při spuštění se v dolní části perspektivy zobrazí pohled Console, který ukazuje výstup spuštěného programu, pokud dojde při překladu k chybám, zobrazí se v pohledu *Problems*.



Obrázek 6.7: Dialogové okno pro kompilaci a spuštění

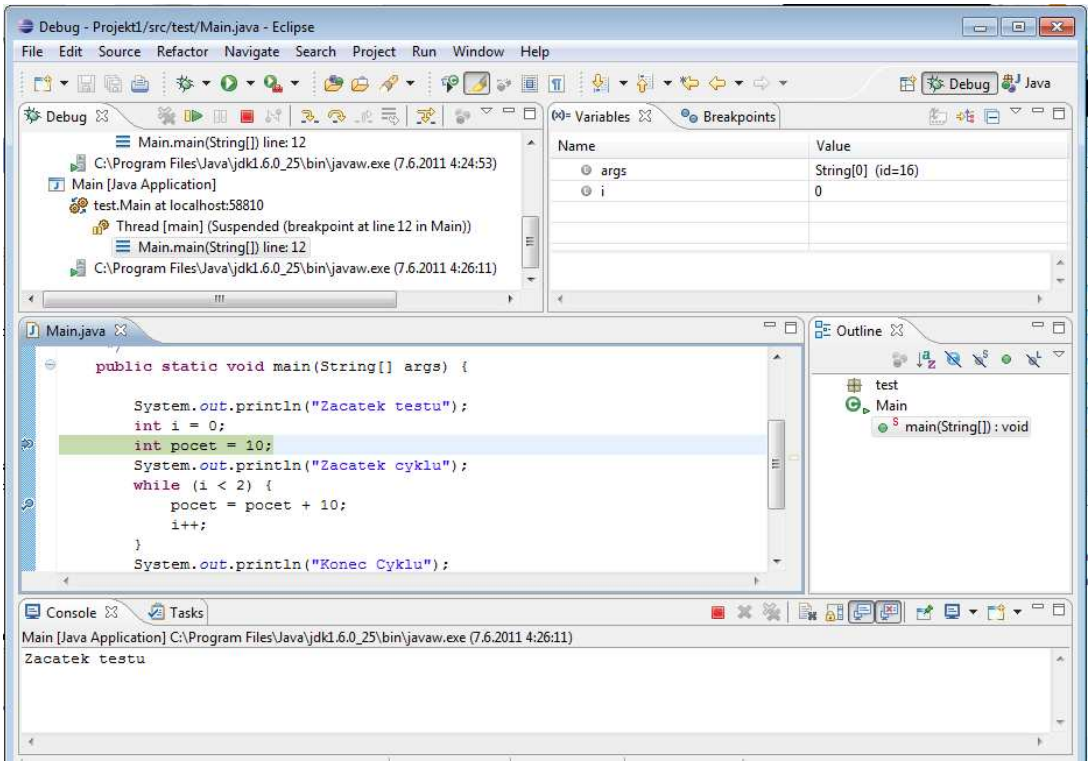
6.7 Debugger

Debbuger je nástroj, který slouží k ladění programů a sleduje stav programu a jeho částí při běhu programu. S pomocí Debuggeru není již nutné používat kontrolní výpisy pro zjištění hodnot proměnných. Umožňuje využití *krokování* a *breakpointů*. *Krokování* je procházení programu po jednotlivých příkazech, *breakpointy* byly vysvětleny už v předešlých kapitolách. *Breakpointy* lze nastavit dvojím poklepáním levého tlačítka myší na řádek, kde se má vykonávání programu zastavit.

Debugger lze spustit pomocí ikony , která je umístěna v nástrojové liště. Také lze spustit Debugger pomocí klávesové zkratky *F11*, v obojím případě se spustí Debugger aktivní třídou. Pomocí *Debug As* se vybere obdobně, jako v případu pouhé kompilace, tedy opět *Java Application*. Po spuštění Debbugeru se změní uspořádání perspektivy Eclipse a zobrazení některých oken (Obrázek 6.8:). Okno editoru se přesunulo do prostřední části pod okno Debug, které zobrazuje všechny spuštěné aplikace. Pod

editorem je okno *Console*, které se nezměnilo z „editační“ perspektivy. V pravé části je *Outline*, které stejně jako v předešle perspektivě znázorňuje průběh programu.

Nový pohled je *Variables*, který zobrazuje proměnné a jejich hodnoty, které se mění v průběhu vykonávání programu. Zobrazují se pouze lokální proměnné. Pro zobrazení konstant a statických proměnných, je nutné zvolit v menu Java možnost *Show Constants*, případně *Show Static Variables*. Okno *Breakpoints* ukazuje umístění *breakpointů* v kódu.



Obrázek 6.8: Debugger Perspective

Debugger lze ovládat pomocí *Nástrojové lišty debuggeru* (Obrázek 6.9). Z levé strany jsou tlačítka této lišty následující: *Remove All Terminated Launches* (vymazání všech aplikací označených jako terminated), *Resume* (posunutí na další breakpoint), *Suspend* (pozastavení běhu programu), *Terminate* (ukončení běhu programu), *Disconnect* (oddělení debugger od vybraného procesu), *Step Into* (krokování), *Step Over* (přeskočení metody), *Step Return* (vrácení na původní úrovni), *Drop To Frame* (znovuotevření vybraného pohledu) a *Use Step Filters* (změna filtrů)

Obrázek 6.9: Nástrojová lišta debuggeru

6.8 Práce s editorem kódu pro Javu

Editor kódu pro programovací jazyk Java obsahuje mnoho užitečných nástrojů, které usnadňují práci. Některé z těchto nástrojů budou podrobněji uvedeny dále. Prvním takovým nástrojem je *Content Assist* (Asistent obsahu), který nabízí různé možnosti automatického doplnění kódu. Vyvolá se klávesovou zkratkou *Crtl + mezerník*. Nastavení tohoto nástroje je v menu *Edit* pod možností *Content Assist*. Funkce *Content Assist on anonymous classes* vygeneruje tělo anonymní třídy i s nezbytnými metodami. *Content Assist* může pomocí *Inserting and replacing in code assit* nahradit identifikátor v kódu.

Dalším nástrojem je automatické doplňování importů, které jsou nutné pro programování projektu. Dalším nástrojem je *Plovoucí nápověda*, která se zobrazí po najetí kurzoru myši na argument metody. Klávesová zkratka je *Ctrl + Shift + Mezerník*. Další nástroj *Templates* (Návrhový vzor) nabízí vzory různých programových struktur, které jsou definované v menu *Window* v možnosti *Preference*s. *Local rename* usnadňuje přejmenování všech výskytů vybraného identifikátoru v aktuálním souboru. Stačí najet kurzorem na identifikátor a pomocí klávesové zkratky se vyvolají možnosti výběru pro záměnu. Eclipse umožňuje odstranění konstrukcí, které obklopují část kódu. Slouží pro to *Remove surrounding statement*.

7 NASTAVENÍ PROSTŘEDÍ PRO PRÁCI V C/C++

Pro vývoj aplikací v programovacím jazyce C/C++ má uživatel možnost použít Eclipse IDE for C/C++ Developers nebo vložit plugin Eclipse *CDT* (C/C++ Development Tooling) do libovolné edice Eclipse. Tento plugin je ke stažení z *http://www.eclipse.org/cdt/*. Pluginy lze do Eclipse integrovat manuálně nebo nástrojem pro instalaci pluginů v Eclipse, což bude popsáno v kapitole 8.1. Následný popis nastavení a konfigurací se vztahuje k verzi Eclipse 3.6 Helios na platformě Microsoft Windows 7. [32]

7.1 Počáteční nastavení pro programovací jazyk C/C++

Aby bylo možné kompilovat C a C++ zdrojové kódy v Eclipse, je potřeba nainstalovat C/C++ *kompilátor* pro Windows. Je dostupné více *kompilátorů*, ale vývojáři Eclipse je doporučováno *MinGW*. *MinGW* také obsahuje hlavičkové soubory a knihovny potřebné pro vývoj aplikací. *MinGW* lze stáhnout z adresy *http://sourceforge.net/projects/mingw/*. Po stažení je nutné *MinGw* nainstalovat do libovolné složky, např. *C:\Program Files\MinGW*.

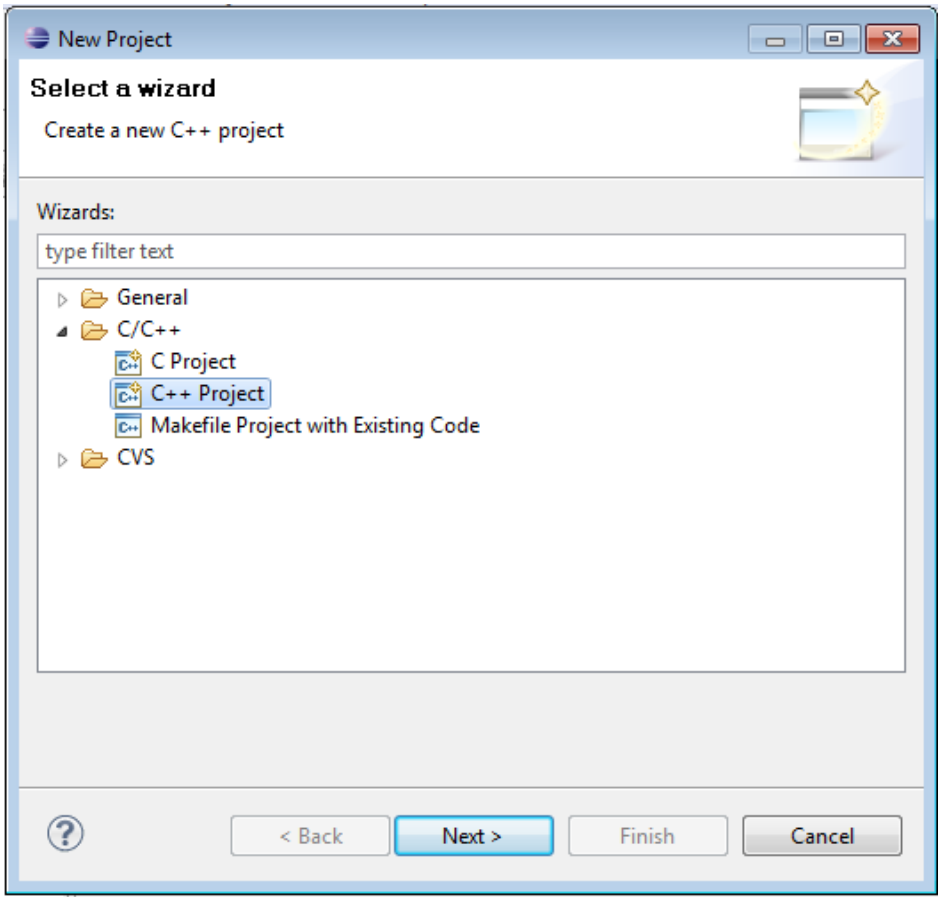
Dále se musí cesta k *MinGW* nastavit do systémové proměnné *PATH* ve Windows. To se provede v menu *Vlastnosti systému*, do kterého se dostane uživatel přes tlačítko START a v nabídce se zvolí možnost *Ovládací panely*. Dále se přejde do menu *Systém a zabezpečení* a pak *Systém*, kde se zvolí *Upřesnit nastavení přístupu*. V menu *Vlastnosti systému* v záložce *Upřesnit* je potřeba stisknout tlačítko *Proměnné prostředí…* . Zde je již možné do proměnné *PATH* vložit cestu k *MinGW*. Nyní je možné v Eclipse kompilovat C/C++ aplikace.

Před samotným začátkem programování si uživatel může nastavit Editor v menu *Windows > Preferences > C/C++ > Editor*. Zde je možné nastavit, např. číslování řádku editoru nebo zvýrazňování závorek. Uživatel si také může nastavit automatické doplňování obsahu v podmenu *Content-Assist*.

7.2 Vytvoření jednoduché aplikace

Pro vytvoření aplikace je nejprve nutné vytvořit nový projekt. V menu *File > New* se vybere položka *Project*. Dále se specifikuje typ nového projektu (Obrázek 7.1). V tomto případě je potřeba vybrat v menu *C++* položku *C++ Project* a kliknout na tlačítko *Next*.

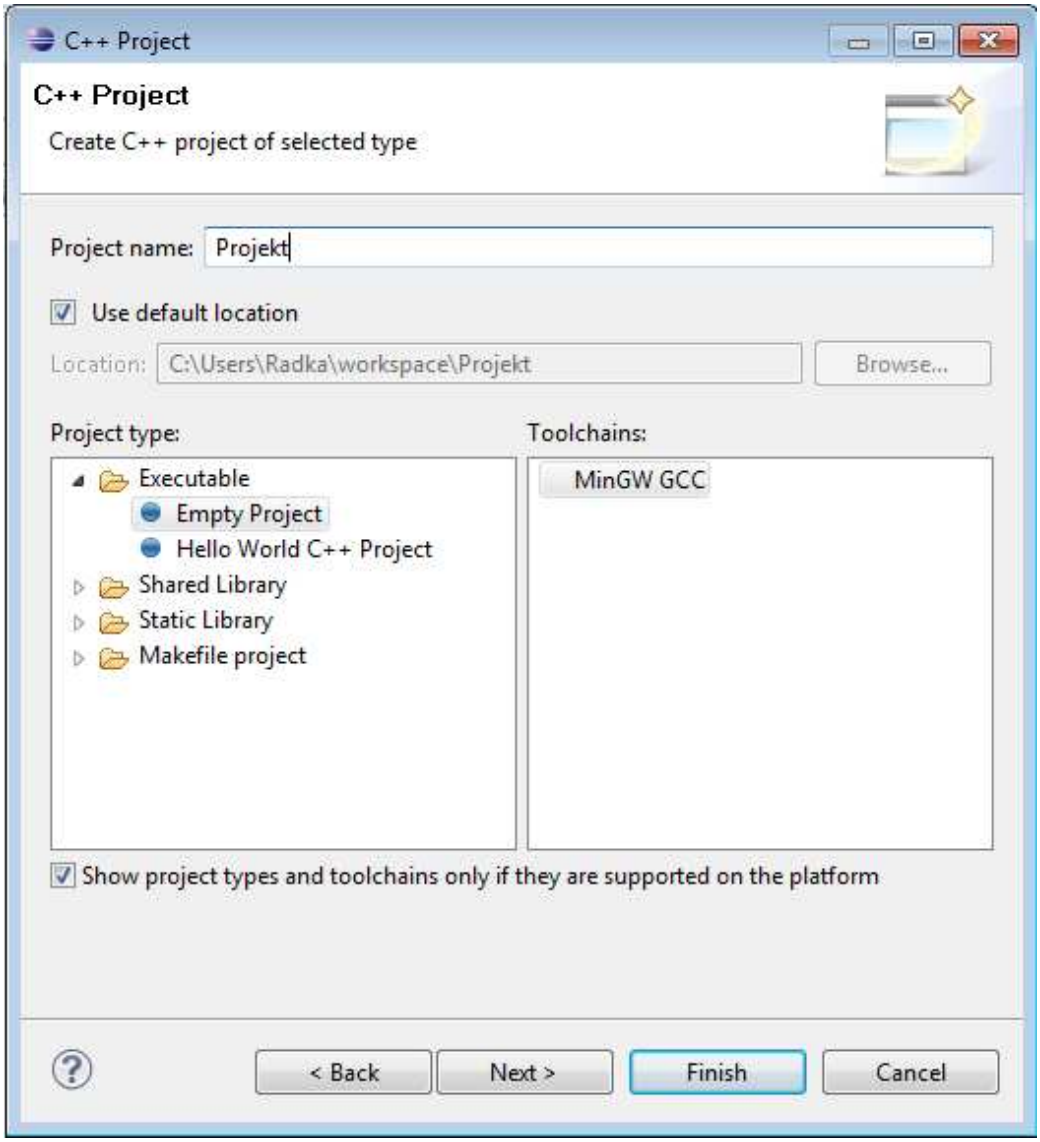
V dalším menu se nastaví parametry projektu (Obrázek 7.2). V poli *Project name* se nastaví jméno projektu. Pokud je *MinGW* správně nastaveno, tak v poli *ToolChain* bude položka *MinGW GCC*, kterou je nutné zvolit. Pokud je nainstalováno více *kompilátorů*, bude možné v tomto poli vybrat i jiné kompilátory. V poli *Project types* v menu *Executables* lze zvolit šablonu k projektu, např. *Hello World C++ Project*, který již obsahuje jednoduchou C++ aplikaci, ze které může uživatel vycházet.



Obrázek 7.1: Okno New Project

V dalším kroku si uživatel může vybrat různé konfigurace projektu. V tomto kroku lze nastavit podrobnější nastavení projektu pomocí tlačítka *Advanced Settings*. V tomto menu je možné nastavit parametry kompilátoru nebo cesty ke knihovnám.

V následujícím kroku je možné nastavit další parametry projektu, kterými jsou, např. jméno autora, licenci a složka se zdrojovými soubory. Pokud uživatel vybral *Hello World C++ Projekt*, tak bude moci také nastavit text, který bude vypsán do konzole. Kliknutím na tlačítko *Finish* je vytvoření projektu s jednoduchou aplikací dokončeno.



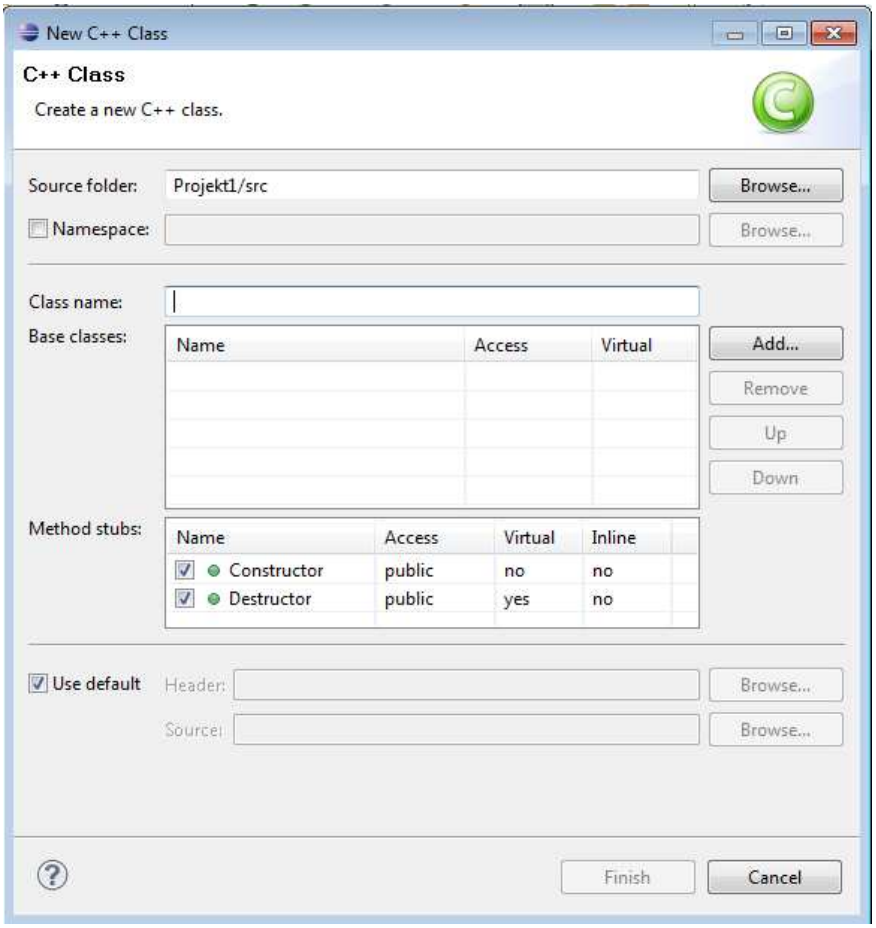
Obrázek 7.2: Vytvoření projektu v C++

7.3 Import existujícího projektu

Pokud je potřeba importovat zdrojové kódy již existující aplikace do Eclipse, je nutné také vytvořit nový projekt. Stejně jako v předcházející kapitole se zvolí *C/C++ Project*. Do pole *Project name* je možné zvolit libovolné jméno projektu. Dále se musí odškrtnout *Use default location* a do pole *Location* vložit cestu k aplikaci, kterou je potřeba importovat. V poli *Projekt Type* je třeba zvolit *Makefile Projekt*, vybrat typ *Empty Project* a pokračovat k dalšímu kroku tlačítkem *Next*. V dalším kroku si uživatel zvolí konfiguraci projektu, stejně jako v předchozí podkapitole. Stiskem tlačítka *Finish* je import existujícího projektu do Eclipse dokončen. V *Průzkumníku* se objeví nový projekt, který obsahuje kódy importované aplikace.

7.4 Vytvoření nového souboru

Vytvoření nových souborů v projektu je podobné jako v kapitole 6.5. Opět lze nový souboru vytvořit v menu *File* > *New* a typ souboru. Nebo kliknutím pravým tlačítkem myši na název projektu v *Průzkumníku* a poté na položku *New*. Nejdůležitějšími soubory jsou Class soubory s třídami aplikace. Pokud uživatel zvolí typ souboru *Class*, bude tento soubor obsahovat novou třídu. Vybráním souboru typu *Class*, se objeví okno (Obrázek 7.3) s možným nastavením. První je potřeba nastavit, kde bude soubor umístěn, což se dělá v poli *Source folder*. Dále zaškrtnutím políčka *Namespace* je možné třídě přiřadit určitý jmenný prostor. V poli *Class name* bude jméno nové vytvořené třídy. Také lze definovat třídy, které budou nově vytvářenou třídu dědit. V menu *Method Stubs* je možné definovat, které metody budou vygenerovány. V nově vytvářené třídě lze vygenerovat *konstruktor* a *destruktor*. Odškrtnutím *Use default* je možné definovat vlastní jméno pro hlavičku a zdrojový soubor, ve kterém třída bude. Jinak se pro název zdrojového souboru i hlavičky použije název třídy. Stisknutím tlačítka *Finish* bude nová třída vytvořena.



Obrázek: 7.3: Vytvoření nové třídy

Také je možné vytvořit libovolný zdrojový a hlavičkový soubor. Pro vytvoření zdrojového souboru se v menu *File > New* se vybere typ *Source file* a následném menu se specifikuje umístění, název a šablona pro nový zdrojový soubor. Šablona slouží pro definici obsahu vygenerovaného zdrojového souboru. Stejný postup lze použít pro vytvoření hlavičkového souboru.

Pro vložení existujícího souboru může uživatel použít stejný postup jako v kapitole 6.6.

7.5 Kompilace a spuštění programu

Projekt je možné zkompilovat v nabídce *Project* vybráním položky *Build Project*. V tomto případě se zkompiluje aktivní projekt. Pokud uživatel vybere *Build All*, tak se zkompilují všechny projekty. V menu *Project* je také možné zaškrtnout položku *Build Automatically* a potom se projekt bude kompilovat automaticky, v případě že se změní zdrojový soubor. Pro rychlou kompilaci je také možné použít symbol kladívka…. v liště nástrojů.

Pro spuštění projektu slouží symbol *Play…* v liště nástrojů nebo je možné v menu *Run* zvolit položku *Run*. Pokud je nastaveno více konfigurací pro spuštění aplikace, tak uživatel zvolí v metu *Run* položku *Run As* a vybere požadovanou konfiguraci. Konfigurace je možné specifikovat v menu *Run > Run Configurations*, kde je možné přiřadit, např. argumenty při spuštění aplikace.

7.6 Debugger

S debuggerem se pracuje stejně jako v případě ladění Java projektů, což bylo popsáno v kapitole 6.7.

8 NASTAVENÍ PROSTŘEDÍ PRO VÝVOJ WEBOVÝCH A SÍŤOVÝCH APLIKACÍ

Pro vývoj webových aplikací je nutné použít Eclipse for Java EE Developers. Opět lze stáhnout samostatné vývojové prostředí nebo doinstalovat do jiné verze pomocí pluginů. Tento tutoriál je vytvářen v Eclipse IDE for Java EE Developers na platformě Windows 7. [32, 34]

8.1 Přidání pluginu do Eclipse

Toto nastavení platí pro všechny standardní pluginy pro vývojové prostředí Eclipse. V této podkapitole je popsáno přidání *Eclipse Web Tools Platform*, který je potřeba nainstalovat pro vývoj webových aplikací. Usnadňuje jejich vývoj a poskytuje pro ně běhové prostředí. Také usnadňuje spuštění webových služeb, jejich umístění na server a ladění webových aplikací. Tento plugin lze nainstalovat v menu *Help* pomocí možnosti *Install New Software…* . V poli *Work with* je nutné vybrat internetovou stránku *Helios - http://download.eclipse.org/releases/helios*. Po zobrazení všech nabídek pluginů, je potřeba vybrat následující možnost *Web, XML, and Java EE Development* a zvolit *Next*.

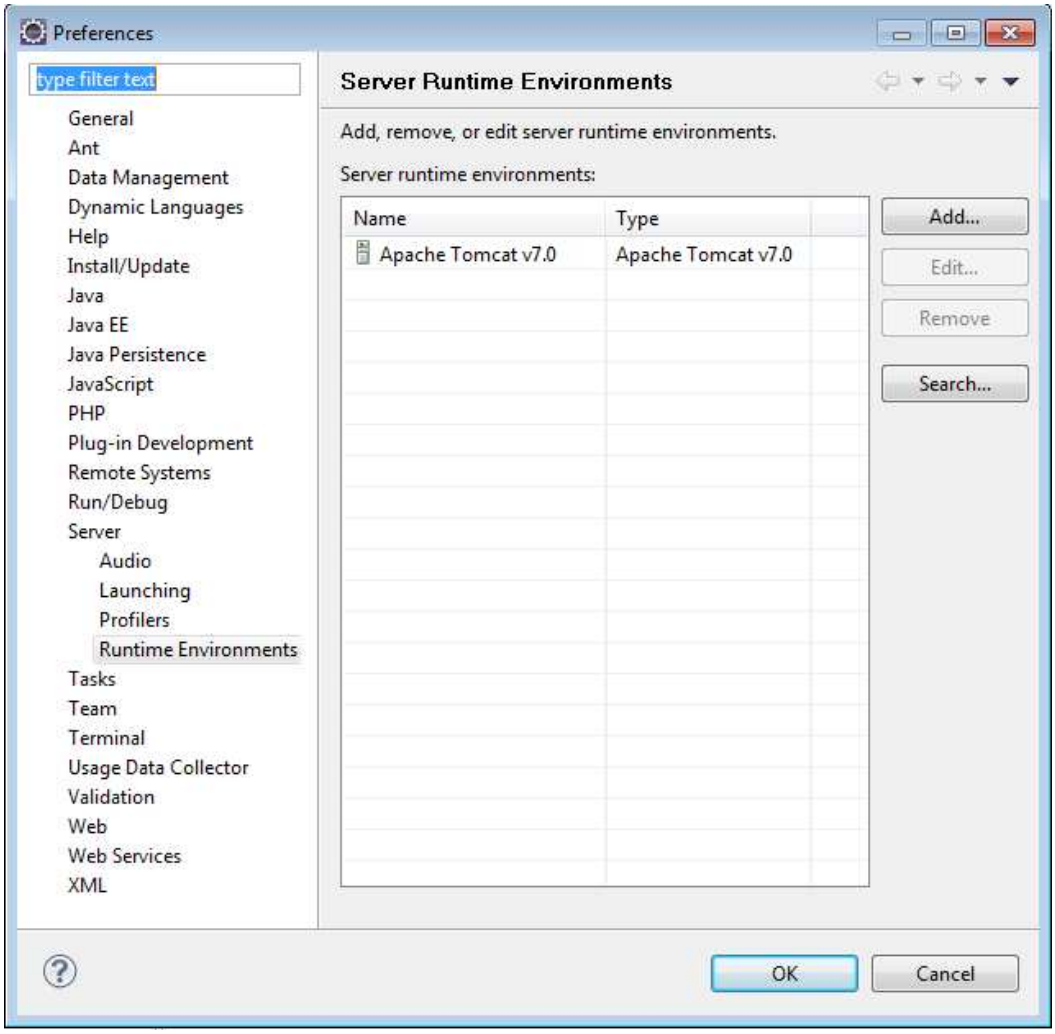
8.2 Nastavení prostředí pro webové aplikace

Pro vývoj webových aplikací v Javě je potřeba nainstalovat nejdříve webový kontejner. V tomto manuálu bude popsán *Apache Tomcat 7.0*, který lze stáhnout z *http://tomcat.apache.org/download-70.cgi*, kde je potřeba vybrat distribuci *32 - bit/64 – bit Windows Service Installer*. Pomocí této distribuce se nainstaluje *Apache Tomcat* jako služba v Microsoft Windows. Po spuštění instalátoru si uživatel může nastavit komponenty, které chce nainstalovat. Poté v menu konfigurace si uživatel může nastavit port, na kterém bude server očekávat HTTP požadavky (standardně 8080). Také je potřeba, aby uživatel nastavil jméno a heslo pro administraci serveru. Poté je možné server nainstalovat.

8.3 Nastavení Runtime Environment

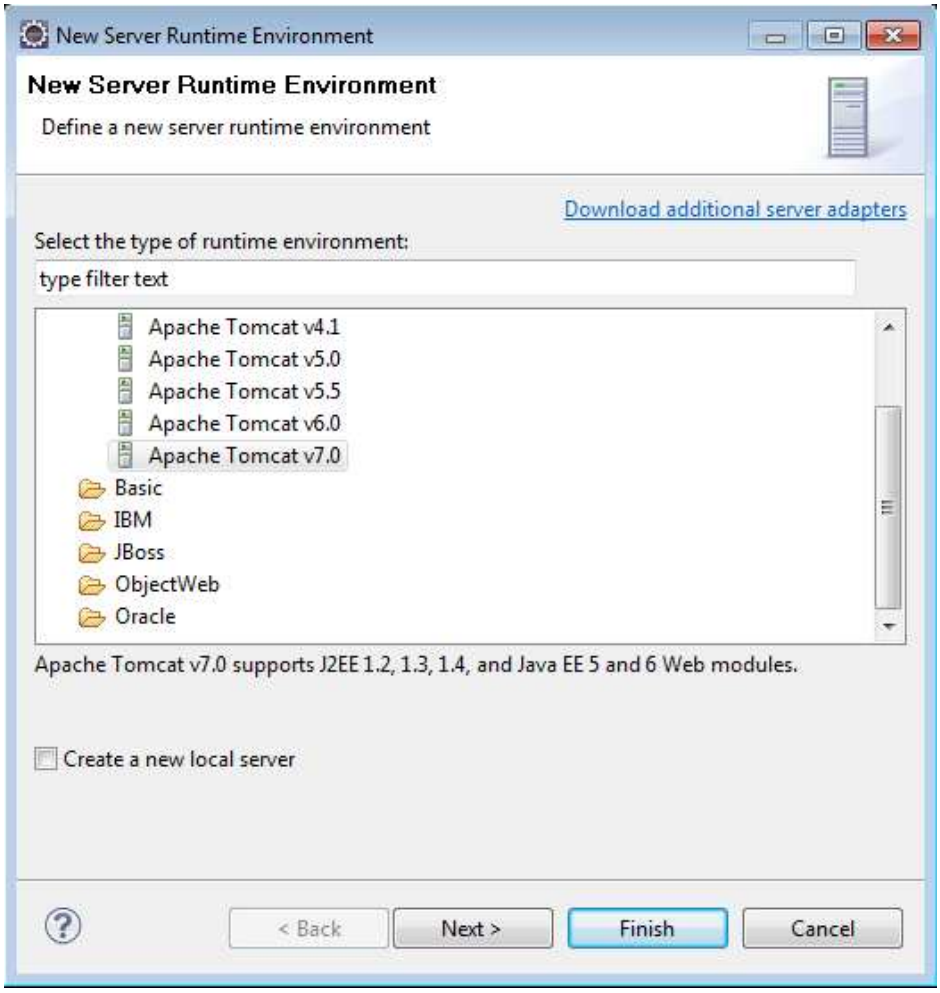
Nejprve je potřeba vytvořit běhové prostředí, ve kterém bude webová aplikace spuštěna. Vybere se okno *Window > Preference > Server > Runtime Environments*, kde je možné

vložit běhové prostředí (Obrázek 8.1). Je potřeba zvolit *Add…*, v novém okně lze vybrat aplikační server. V tomto případě se zvolí *Apache Tomcat v7.0*, ale je možné také vybrat *JBoss* nebo *IBM WebSphere*, záleží na nainstalovaných běhových prostředí.



Obrázek 8.1: Okno pro přidání serveru

V dalším kroku se zobrazí okno *New Server Runtime Environment* (Obrázek 8.2), ve kterém se vybere název běhového prostředí a cesta k jeho umístění. Poté pomocí tlačítka *Finish* se dokončí konfigurace běhového prostředí. Dále je potřeba přiřadit server k běhovému prostředí pomocí okna, které se zobrazí pomocí menu *Window* > *Show View* > *Servers* > *Servers*. V tomto okně se klikne kdekoliv pravým tlačítkem myši a z menu se vybere *New* > *Server*. Zde si uživatel vybere server, kterému je potřeba přiřadit *Host name*, jméno serveru a běhové prostředí, které bylo vytvořeno v předešlém kroku.



Obrázek 8.2: Okno New Server Runtime Environment

8.4 Vytvoření jednoduché webové aplikace

Pro vytvoření nové webové aplikace je potřeba kliknout na *Add New Project* > *Web* > *Dynamic Web Project*. V dalším kroku se zobrazí okno *New Dynamic Web Project*, kde si uživatel nastaví jméno projektu, v poli *Target runtime* se přiřadí projektu webové prostředí. V poli *Configuration* lze vybrat konfiguraci běhového prostředí. Po kliknutí na Next se zobrazí okno, kde je možné vybrat složky, ve kterých budou zdrojové a přeložené třídy. Po kliknutí na *Finish* se vytvoří nový projekt webové aplikace.

Projekt prozatím prázdný, proto je potřeba přidat nebo vytvořit zdrojové soubory. Do projektu bude přidána *JSP* (*Java Server Pages*) stránka. JSP stránky umožňují snadný vývoj webového dynamického obsahu. JSP je složená z *tagů*, nejčastěji jazyka HTML a *scripletů*, což jsou osamocené části kódu Java, které generují dynamický obsah. Pomocí pravého tlačítka na složku *WebContent* v *Project Explorer* se vyvolá menu, ve kterém

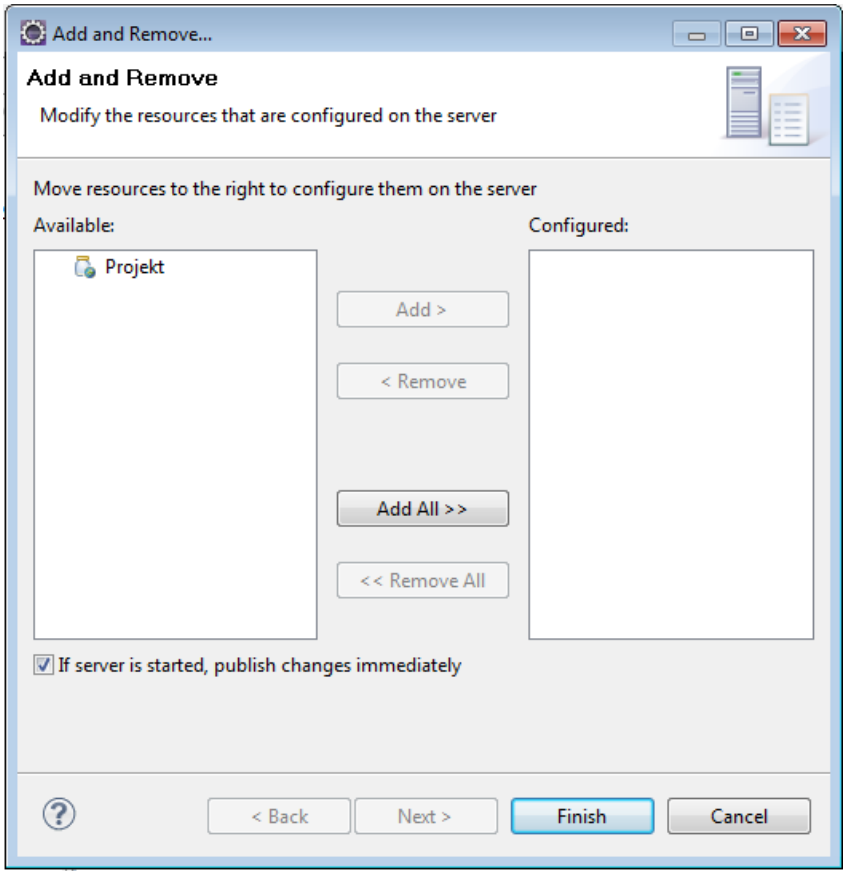
uživatel zvolí možnost *New* > *JSP File*. V následujícím okně vyplní jméno souboru a zvolí *Next*. V dalším kroku vybere typ JSP souboru a zvolí *Finish*. Tento soubor je po vytvoření prázdný a je potřeba doplnit kód. Testovací JSP soubor je v Příloze III.

8.5 Překlad projektu

Překlad projektu se provádí stejně jako v kapitole 6.6.

8.6 Spuštění projektu

Pro spuštění samotného projektu je nutné rozběhnout server. Přehled serverů je v okně *Servers*, kde uživatel zvolí požadovaný server, pravým tlačítkem myši se na něj klikne a z menu zvolí *Start*. Poté se musí počkat, dokud server nenaběhne. Po spuštění serveru je nutné aplikaci umístit na server. Opět se zvolí server a tentokrát možnost *Add and Remove…* . Zde v poli *Available* je seznam aplikací, které je možné umístit na server. Uživatel zvolí svoji aplikaci a pomocí tlačítka *Add* ji přesune do pole *Configured*, po klinutí na *Finish* je aplikace umístěna na server.



Obrázek 8.3: Přidání / Odebrání aplikace na server

Nyní je možné aplikaci otestovat. Do webového prohlížeče se vloží URL () ve formátu *http://localhost:8080/<název projektu>/<název JSP souboru>*. Pokud je vše dobře nastaveno, zobrazí se v prohlížeči www stránka vygenerovaná z JSP souboru.

8.7 Debugger

Při ladění aplikace je nutné server spustit v debuggovacím režimu. Opět v okně *Servers* se vybere server, na kterém se bude aplikace ladit. Uživatel klikne pravým tlačítkem myši a vybere možnost *Debug*, čímž se server spustí v debuggovacím režimu. Dále je možné postupovat stejně jako v kapitole 6.7.

9 TVORBA VIDEOTUTORIÁLU

Pro tvorbu video tutoriálu byl vybrán program CamStudio (Obrázek č. 9.1). Má hned několik výhod. Jedná se o jednoduchý program, který přesto zvládá poměrně hodně funkcí. Je to open source, který se je pod licencí GPL, proto jsou všechny jeho funkce dostupné. Samotný program není velký a nezabírá hodně místa ani po instalaci. Podporuje nahrávání videa ve více formátech.

Obrázek č. 9.1: CamStudio:

9.1 Popis nastavení nahrávání

Ovládání programu je jednoduché. V horní liště se nastavují různé vlastnosti snímání obrazovky. Spodní lišta obsahuje základní prvky ovládání. Nahrávání lze spustit, buď červeným kolečkem, pomocí menu *File* > *Record*, nebo pomocí klávesové zkratky v menu *Options* > *Keyboard Shortcuts*. Dalším nastavením je plocha snímání videa, lze nastavit snímání celé plochy (*Full Screen*) nebo výběr určité oblast (*Region*). Tento program také podporuje snímání Autopan, což je možnost snímaní části obrazovky pohybem kurzoru. V CamStudiu lze nahrát zvuk z mikrofonu nebo z výstupu zvukové karty. Užitečným nástrojem je také *Srceen Annotations*, který umožňuje vytvářet různobarevná textová pole. Kvalita snímání se nastavuje v menu *Options*. [31]

ZÁVĚR

Cílem této bakalářské práce bylo vytvořit video tutoriál pro vývojové prostředí Eclipse. Jedním z bodů zadání bylo také porovnat nejnovější verzi IDE Eclipse 3.6 Helios s ostatními vývojovými prostředími. V první kapitole bylo popsáno Microsoft Visual Studio 2010 a v druhé kapitole IDE NetBeans. Práce popisuje všechny tři prostředí podrobněji, aby si čtenář vytvořil názor na tato prostředí sám. Celkové shrnutí jejich porovnání je uvedeno v kapitole čtyři.

Popis vývojového prostředí Eclipse je popsán v páté kapitole. Jsou zde uvedeny pouze základní nástroje, protože Eclipse disponuje mnoha nástroji, které lze přidat pomocí pluginů. Tato práce se snaží pomoci začínajícím programátorům v pochopení vývojového prostředí.

V šesté, sedmé a osmé kapitole jsou popsány návody pro práci s jednotlivými programovacími jazyky. Nejdříve je popsáno základní nastavení pro programovací jazyk Java, protože Eclipse tento jazyk primárně podporuje. Další nastavení jsou pro jazyky C a C++, protože je Eclipse v dnešní době považováno za jeden z nejlepších prostředí pro tyto jazyky. Poslední nastavení je pro vývoj webových aplikací.

Vhodným rozšířením této práce by bylo postupné přidání návodů dalších programovacích jazyků nebo návody pro vývoj aplikací na další platformě, případně samotná tvorba pluginů pro náročnější uživatele.

ZÁVĚR V ANGLIČTINĚ

The aim of this thesis was to create a video tutorial on IDE Eclipse. One of the points of the assignment was to compare the latest Eclipse IDE version 3.6 Helios with other development environments. The first chapter describes Microsoft Visual Studio 2010 and the second chapter focuses NetBeans IDE. This work studies in detail on all three environments so that reader can create his own opinion on these environments themselves. In chapter four is summary all IDEs.

Description of the Eclipse development environment is in chapter five. There are only basic Eclipse tools; Eclipse has many tools that can be added via plugins. This work aims to help novice programmers in understanding the development environment.

The sixth, seventh and eighth chapter describes guidelines for working with different programming languages. First, it describes the basic settings for the Java programming language, because Java is primary language for Eclipse. Other settings are for C and C + +, because Eclipse is today considered one of the best environments for these languages. The last setting is for web development.

Convenient extension of this work could be another addition of manuals of other programming languages or manuals for developing applications on other platforms.

SEZNAM POUŽITÉ LITERATURY

[1] *MSDN Česká republika: Microsoft Visual Studio* [online]. c2011 [cit. 2011-06-01]. Dostupné z WWW: <http://www.microsoft.com/cze/msdn/vstudio/2010/>.t

[2] Microsoft Visual Studio. In *Wikipedia : the free encyclopedia* [online]. St. Petersburg (Florida) : Wikipedia Foundation, 7. 4. 2009, last modified on 26. 5. 2011 [cit. 2011-06-01]. Dostupné z WWW:

<http://en.wikipedia.org/wiki/Microsoft_Visual_Studio>.

[3] Microsoft Visual Studio. In *Wikipedia : the free encyclopedia* [online]. St. Petersburg (Florida) : Wikipedia Foundation, 21. 3. 2006, last modified on 2. 5. 2011 [cit. 2011-06-01]. Dostupné z WWW:

<http://cs.wikipedia.org/wiki/Microsoft_Visual_Studio>.

[4] *MSDN Česká republika* [online]. c2011 [cit. 2011-06-01]. Porovnání a ceny. Dostupné z WWW:

<http://www.microsoft.com/cze/msdn/vstudio/porovnani.aspx>.

[5] *MSDN : Library Visual Studio 2010* [online]. c2011 [cit. 2011-06-01]. Dostupné z WWW: <http://msdn.microsoft.com/en-us/library/dd831853.aspx>.

[6] *What's New in Visual Studio 2010* [online]. c2011 [cit. 2011-06-01]. MSDN. Dostupné z WWW: <http://msdn.microsoft.com/en-us/library/bb386063.aspx>.

[7] *Produktová řada Microsoft Visual Studio 2010* [online]. c2011 [cit. 2011-06-01]. MSDN Česká republika. Dostupné z WWW: <http://www.microsoft.com/cze/msdn/vstudio/2010/>.

[8] HALVORSON, Michael. *Microsoft Visual Basic 2010 : krok za krokem*. 1. vydání, Praha : Computer Press, 2010. 480 s. ISBN 978-80-251-3146-6, K1835.

[9] *NetBeans* [online]. c2011 [cit. 2011-06-01]. Dostupné z WWW: <http://netbeans.org/index.html>.

[10] *COMMON DEVELOPMENT AND DISTRIBUTION LICENSE*. [s.l.] : Oracle, 2004. Dostupné z WWW: <http://www.sun.com/cddl/cddl.html>.

[11] *GNU General Public License, version 2*. 51 Franklin Street, Fifth Floor, Boston, MA 02110-1301, USA : Free Software Foundation, Inc. , červen 1991 . Dostupné z WWW: <http://www.gnu.org/licenses/gpl-2.0.html>.

[12] NetBeans. In *Wikipedia : the free encyclopedia* [online]. St. Petersburg (Florida) : Wikipedia Foundation, 21 April 2003, last modified on 13. 5. 2011 [cit. 2011-06-01]. Dostupné z WWW: <http://en.wikipedia.org/wiki/NetBeans>.

[13] Vývojová platforma NetBeans. In *Wikipedia : the free encyclopedia* [online]. St. Petersburg (Florida) : Wikipedia Foundation, 18. 3. 2010, last modified on 20. 12. 2010 [cit. 2011-06-02]. Dostupné z WWW: <http://cs.wikipedia.org/wiki/V%C3%BDvojov%C3%A1_platforma_NetBeans>.

[14] NetBeans. In *Wikipedia : the free encyclopedia* [online]. St. Petersburg (Florida) : Wikipedia Foundation, 21. 12. 2004, last modified on 13. 5. 2011 [cit. 2011-06-01]. Dostupné z WWW: <http://cs.wikipedia.org/wiki/NetBeans>.

[15] BÖCK, Heiko. *Platforma NetBeans : Podrobný průvodce programátora*. 1. Brno : Computer Press a.s., 2010. 320 s. ISBN 978-80-251-3116-9, K1828.

[16] *Eclipse - The Eclipse Foundation open source community webside* [online]. c2011 [cit. 2011-06-01]. Eclipse. Dostupné z WWW: <http://www.eclipse.org/>.

[17] *Eclipse* [online]. c2011 [cit. 2011-06-01]. Eclipse Public License - v 1.0. Dostupné z WWW: <http://www.eclipse.org/legal/epl-v10.html>.

[18] Eclipse (vývojové prostředí). In *Wikipedia : the free encyclopedia* [online]. St. Petersburg (Florida) : Wikipedia Foundation, 29. 1. 2006, last modified on 14. 3. 2011 [cit. 2011-06-06]. Dostupné z WWW: <http://cs.wikipedia.org/wiki/Eclipse_%28v%C3%BDvojov%C3%A9_prost%C5%99ed%C3%AD%29>.

[19] Eclipse (software). In *Wikipedia : the free encyclopedia* [online]. St. Petersburg (Florida) : Wikipedia Foundation, 26. 4.2003 , last modified on 4 June 2011 [cit. 2011-06-06]. Dostupné z WWW: <http://en.wikipedia.org/wiki/Eclipse_%28software%29>.

[20] ERL, Thomas. *Service-Oriented Architecture (SOA): Concepts, Technology, and Design*. Boston : Prentice Hall, 2005. 792 s.

[21] *Eclipse - Mylyn Open Source : Mylyn* [online]. c2011 [cit. 2011-06-01]. Dostupné z WWW: <http://www.eclipse.org/mylyn/>.

[22] *Microsoft Visual Studio 2010* [online]. c2011 [cit. 2011-06-01]. Windows Forms. Dostupné z WWW: <http://msdn.microsoft.com/cs-cz/library/dd30h2yb.aspx>.

[23] *ASP.NET Developer Centre* [online]. c2011 [cit. 2011-06-01]. Dostupné z WWW: <http://msdn.microsoft.com/en-GB/asp.net/>.

[24] *Java.net* [online]. 2008 [cit. 2011-06-01]. JAX-WS Reference Implementation. Dostupné z WWW: <http://jax-ws.java.net/>.

[25] *Eclipse* [online]. c2011 [cit. 2011-06-01]. Web Tools Platform Project. Dostupné z WWW: <http://www.eclipse.org/webtools/>.

[26] *Eclipsepedia* [online]. 14.12. 2005 [cit. 2011-06-01]. JFaces. Dostupné z WWW: <http://wiki.eclipse.org/index.php/JFace>.

[27] *Microsoft WindowsClient.NET* [online]. c2011 [cit. 2011-06-01]. WPF and Silverlight Designer for Visual Studio 2010. Dostupné z WWW: <http://windowsclient.net/wpfdesigner/>.

[27] *MSDN* [online]. c2011 [cit. 2011-06-01]. XAML Overview. Dostupné z WWW: <http://msdn.microsoft.com/en-us/library/ms752059.aspx>.

[28] *MSDN* [online]. c2011 [cit. 2011-06-01]. LINQ Framework Development center. Dostupné z WWW: <http://msdn.microsoft.com/en-us/netframework/aa904594>.

[29] *Oracle* [online]. c2011 [cit. 2011-06-01]. Java SE Downloads. Dostupné z WWW: <http://www.oracle.com/technetwork/java/javase/downloads/index.html>.

[30] *COMPUTERWORLD* [online]. 14.3.2011 [cit. 2011-06-01]. Nejlepší nástroje v programování v Javě. Dostupné z WWW: <http://computerworld.cz/vyvoj/nejlepsi-nastroje-na-programovani-v-jave-42939>.

[31] *SWMAG.cz* [online]. 21. 3. 2010 [cit. 2011-06-01]. CamStudio - zachytávání videa z obrazovky zdarma. Dostupné z WWW: <http://www.swmag.cz/613/camstudio-zachytavani-videa-z-obrazovky-zdarma/>.

[32] *Help Eclipse Helios* [online]. c2011 [cit. 2011-06-01]. Eclipse documentation - Current Release . Dostupné z WWW: <http://help.eclipse.org/helios/index.jsp>.

[33] *Eclipse* [online]. 2007 [cit. 2011-06-01]. Dostupné z WWW: <http://home.zcu.cz/~housky/Skola/eclipse.pdf>.

[34] *Vogella.de* [online]. 6.12.2010 [cit. 2011-06-01]. Servlet and JSP development with Eclipse WTP . Dostupné z WWW: <http://www.vogella.de/articles/EclipseWTP/article.html#installation_tomcat>.

SEZNAM POUŽITÝCH SYMBOLŮ A ZKRATEK

.NET	.
API	Application Interface.
ASP	Active Server Pages.
AWT	Abstract Window Toolkit.
CDDL	Common Development and Distribution License.
COM	Component OBject Model.
CSS	Cascading Style Sheets.
CVS	Concurrent Version System.
EMF	Eclipse Modeling Framework.
EMF	Eclipse Modeling Framework.
EMFT	Eclipse Modeling Framework Technology.
EPL	Eclipse Public License.
GMF	Graphical Modeling Framework.
GMF	Graphical Modeling Framework.
GPLv2	GNU General Public License version 2.
GUI	Graphic User Interface.
HTML	HyperText Markup Language.
IDE	Integrated Development Enviroment.
J2EE	Java 2 Enterprise Edition.
J2ME	Java 2 Micro Edition.
J2SE	Java 2 Standard Edition.
JDK	Java Development Kit.
JDT	Java Development Tools.
JPA	Java Persistence API.

JSF	JavaServer Faces.
M2M	Model To Model.
M2T	Model To Text.
MFC	Microsoft Foundation Class Library.
MPF	Manager Package Framework.
MSDN	Microsoft Developer Network.
MSSCCI	Microsoft Source Code Control Interface.
OSGi	Open Services Gateway initiative.
PDT	PHP Developers Tools.
RCP	Rich Client Platform.
SDK	Softwarw Development Kit.
SQL	Structured Query Language.
Swing	Java GUI widget toolkit.
SWT	The Standard Widget Toolkit.
TDD	Test Driven Development.
TFS	Team Foundation Server.
UML	Unified Modeling Language.
VB.NET	Visual Basic .NET.
WPF Designer	Microsoft Software Developer Network.
XML	Extensible Markup Language.
XSLT	Extensible Stylesheet Language Transformations.

SEZNAM OBRÁZKŮ

SEZNAM TABULEK

SEZNAM PŘÍLOH

PŘÍLOHA P I: TABULKA VLASTNOSTÍ JEDNOTLIVÝCH EDIC VISUAL STUDIA

Verze	***Visual Studio Express***	***Visual Studio Standart***	***Visual Studio Professional***	***Visual Studio Team System***
Rozšíření	*Neobsahuje*	**Ano**	**Ano**	**Ano**
Externí nástroje	Omezené	**Ano**	**Ano**	**Ano**
Nastavení projektů	Omezené	Omezené	**Ano**	**Ano**
MSDN integrace	Funkce není součástí, nutné doinstalovat	**Ano**	**Ano**	**Ano**
Designer tříd	*Neobsahuje*	**Ano**	**Ano**	**Ano**
Refaktorování	Omezené	**Ano**	**Ano**	**Ano**
Debugging	Omezené	**Ano**	**Ano**	**Ano**
64-bitové aplikace	*Neobsahuje*	**Ano**	**Ano**	**Ano**
Procesory Itanium	*Neobsahuje*	*Neobsahuje*	*Neobsahuje*	**Ano**
Visual Studio Tools for Office	*Neobsahuje*	*Neobsahuje*	**Ano**	**Ano**

PŘÍLOHA P II: TABULKA NOVINEK A VLASTNOSTÍ VISUAL STUDIA 2010

Vlastnosti a údaje uvedené v této tabulce nejsou kompletní a jsou spíše orientační. Vycházejí ze domácích stránek Visual Studio 2010

Zdroj : http://msdn.microsoft.com/library/bb386063.aspx

Název, novinka	**Popis**
Visual Basic 2010	Obsahuje nové funkce pro jazyk Visual Basic, nový editor kódu, implicitní pokračování řádku, kolekci inicializátorů
Visual C# 2010	Opět obsahuje nové funkce pro jazyk Visual C#, nové prvky editoru kódu, dynamický typ, vylepšenou podporu pro Office
Visual C++ 2010	Nové funkce, včetně vyjádření lambda výrazů,
Visual F# 2010	Podpora jazyka F# a programování pro .NET Framework
Začínáme s Visual Studio	Po spuštění ukáže uživateli, jak používat Visual Studio 2010
Rychlá přehlídka po integrovaném vývojovém prostředí	Stručný přehled o funkcích a nástrojích, které jsou součástí Visual Studia
. NET Framework 4	Obsahuje mnoho nových funkce a vylepšení pro .NET Framework 4 • Kompatibilní s předešlými verzemi • Zvýšeno zabezpečení • Profil klienta podporuje více platforem • Souběžné spouštění v průběhu procesu • Lepší uvolnění paměti

	• Dynamické
Visual Studio Application Lifecycle Management 2010	Nahrazuje Team Foundation Server.
Editor kódu	• Okna editoru už nejsou omezena okraji vývojového prostředí, nyní je lze ukotvit ke krajům prostředí nebo posunout kamkoli po pracovní ploše. • Pokud jsou otevřená dvě okna, změny se projeví okamžitě v obou oknech • Nástroj lupa, funguje pouze v aktuálních oknech, pomocí CTRL + rolovací kolečko myši • Výběr pole – v předešlých verzích byl pouze obdélníkový výběr, ve Visual 2010 je možné použít pole výběru • Zobrazení hierarchie – slouží pro snadnější přehled kódu, je k dispozici pouze pro jazyky Visual C#a Visual C++ • Editor také umožňuje navést uživatele na hledanou část textu pomocí funkce Navigate to • Zobrazí instance symbolu, pokud na ně uživatel klikne. Případně umožňuje posunutí na další symbol. • Umožňuje vygenerovat třídu nebo část třídy před jejím definováním. • Technologie IntelliSence – poskytuje dva druhy nápovědy pro dokončování příkazů, režim dopisování a režim nárvhu.
Správa rozšíření aplikace Visual Studio	Obsahuje přehled možných rozšíření pro danou verzi, online galerii aj

PŘÍLOHA P III: ZDROJOVÝ KÓD PRO JSP SOUBOR

```
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
     pageEncoding="ISO-8859-1"%>
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html>
       <head>
           <meta http-equiv="Content-Type" content="text/html;
charset=ISO-8859-1">
           <title>JSP s aktuálním datem</title>
       </head>

       <body>
            <%java.text.DateFormat df = new
java.text.SimpleDateFormat("dd/MM/yyyy"); %>

           <h1>Aktualni datum: <%= df.format(new java.util.Date())
%></h1>
      </body>

</html>
```

PŘÍLOHA P IV: OBSAH PŘILOŽENÉHO DVD

fulltext.pdf	Text bakalářské práce
videotutorial1_java.wmv	Tutoriál pro nastavení Eclipse v Javě
videotutorial2_c.wmv	Tutoriál pro nastavení Eclipse v C/C++
videotutorial3_web.wmv	Tutoriál pro nastavení Eclipse pro vývoj webových aplikací
videotutorial1_java.avi	Tutoriál pro nastavení Eclipse v Javě
videotutorial2_c.avi	Tutoriál pro nastavení Eclipse v C/C++
videotutorial3_web.avi	Tutoriál pro nastavení Eclipse pro vývoj webových aplikací