ZÁPADOČESKÁ UNIVERZITA V PLZNI

FAKULTA STROJNÍ

Studijní program: B 2341 Strojírenství

Studijní zaměření: Informační a komunikační technologie ve strojírenském podniku

BAKALÁŘSKÁ PRÁCE

Tvorba výukového programu ve Wintermute Engine

Autor: **Tomáš Andrle**

Vedoucí práce: **Ing. Petr Hořejší, Ph.D.**

Akademický rok 2013/2014

Prohlášení o autorství

Předkládám tímto k posouzení a obhajobě bakalářskou práci, zpracovanou na závěr studia na Fakultě strojní Západočeské univerzity v Plzni.
Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně, s použitím odborné literatury a pramenů, uvedených v seznamu, který je součástí této bakalářské práce.

V Plzni dne: ……. **. . . . . .**

podpis autora

ANOTAČNÍ LIST BAKALÁŘSKÉ PRÁCE

AUTOR	**Příjmení** Andrle	**Jméno** Tomáš
STUDIJNÍ OBOR	B2341 „Informační a komunikační technologie ve strojírenském podniku“	
VEDOUCÍ PRÁCE	**Příjmení (včetně titulů)** Ing. Hořejší, Ph.D.	**Jméno** Petr
PRACOVIŠTĚ	ZČU - FST - KPV	

DRUH PRÁCE	~~**DIPLOMOVÁ**~~	**BAKALÁŘSKÁ**	**Nehodící se škrtněte**
NÁZEV PRÁCE	Tvorba výukového programu ve Wintermute Engine		

FAKULTA	strojní		**KATEDRA**	KPV		**ROK ODEVZD.**	2014

POČET STRAN (A4 a ekvivalentů A4)

CELKEM	55		**TEXTOVÁ ČÁST**	41		**GRAFICKÁ ČÁST**	14

STRUČNÝ POPIS (MAX 10 ŘÁDEK) **ZAMĚŘENÍ, TÉMA, CÍL POZNATKY A PŘÍNOSY**	Cílem bakalářské práce je tvorba výukového programu v konceptu digitální továrny, který simuluje dílnu ve strojírenském podniku. Při tvorbě tohoto programu byl použit softwarový balík Wintermute Engine, který je blíže představen v rešeršní části této práce.
KLÍČOVÁ SLOVA **ZPRAVIDLA JEDNOSLOVNÉ POJMY, KTERÉ VYSTIHUJÍ PODSTATU PRÁCE**	digitální továrna, vážná hra, virtuální dílna, Wintermute Engine

SUMMARY OF BACHELOR SHEET

AUTHOR	**Surname** Andrle	**Name** Tomáš	
FIELD OF STUDY	B2341 „Information and Communication Technology in Industrial Management“		
SUPERVISOR	**Surname (Inclusive of Degrees)** Ing. Hořejší, Ph.D.	**Name** Petr	
INSTITUTION	ZČU - FST - KPV		
TYPE OF WORK	~~**DIPLOMA**~~	**BACHELOR**	**Delete when not applicable**
TITLE OF THE WORK	Development of Tutorial Programme in Wnitermute Engine		

FACULTY	Mechanical Engineering	**DEPARTMENT**	Industrial Engineering and Management	**SUBMITTED IN**	2014

NUMBER OF PAGES (A4 and eq. A4)

TOTALLY	55	**TEXT PART**	41	**GRAPHICAL PART**	14

BRIEF DESCRIPTION **TOPIC, GOAL, RESULTS AND CONTRIBUTIONS**	The aim of the thesis is the development of an educational program in the digital factory concept, which simulates a workshop in engineering company. In developing this program was used Wintermute Engine software package, which is introduced in the search section of this paper.
KEY WORDS	digital factory, serious game, virtual workshop, Wintermute Engine

Poděkování

Na tomto místě bych rád poděkoval vedoucímu mé bakalářské práce Ing. Petru Hořejšímu, Ph.D. za cenné připomínky a odborné rady při zpracovávání bakalářské práce. Dále děkuji studentovi Západočeské Univerzity Bc. Jiřímu Polcarovi za zpracování modelu virtuální dílny.
Zároveň bych chtěl poděkovat mojí rodině za trpělivost a podporu po celou dobu mého studia.

Obsah

Seznam obrázků

Seznam tabulek

Seznam zkratek

GNU – GNU's Not Unix!
GPL – General Public Licence
GTK – GIMP Toolkit
HD – high-definition
MP3 – formát ztrátové komprese zvukových souborů
MS1 – Manažerský simulátor 1
MIT – Massachusetts Institute of Technology
OS X – Operační systém firmy Apple
PC – Personal Computer
PHP – Hypertextový preprocesor
SciTE – Scintilla Text Editor
SSH – Secure Shell
VR – Virtuální realita
WME – Wintermute Engine
WYSIWYG – What You See Is What You Get

1 Úvod a cíle práce

Počítače a výpočetní technika jsou v dnešní době chápány jako naprostá samozřejmost, se kterou se denně dostává do styku téměř každý člověk. Mimo provádění složitých matematických výpočtů a zpracovávání ohromného množství dat pomocí databází, ke kterému byly počítače v době svého vzniku předurčeny, se stále častěji využívají k simulaci procesů známých z běžného života, nebo k zábavě při trávení volného času. Spojením těchto funkcí se dostaneme k dalšímu fenoménu dnešní doby, a to jsou počítačové hry. V následujícím textu si předvedeme, že počítačové hry nemusí být pouze „požírač“ volného času, ale že mohou být v mnoha případech velice přínosnou technologií.

Tato bakalářská práce se zabývá tvorbou tzv. vážné hry, nebo jinými slovy výukového programu, který simuluje dílnu fiktivního podniku. Zde se uživatel může seznamovat s přítomnými stroji a způsobem práce na nich. Celá simulace virtuální dílny se odehrává v jedné výrobní hale, kde jsou umístěny tři nejběžnější stroje užívané ve strojírenském podniku. Jmenovitě se jedná o sloupovou vrtačku, svislou konzolovou frézku a univerzální hrotový soustruh. Podrobnějším představením výukového programu se zabývá kapitola 5, kde je popsán postup tvorby simulace.

Primárním cílem této práce je tedy tvorba programu, který poskytne uživateli základní informace z oblasti obrábění kovu. Tento program má za úkol zábavnou formou seznámit uživatele s výše uvedenými stroji. Popisuje jednotlivými části těchto strojů a způsob jejich fungování. V druhé části simulace má uživatel možnost si nově nabyté vědomosti ihned ověřit při simulované výrobě jednoduchého dílu. Touto formou získá základní přehled o běžně využívaných výrobních metodách a pracovních postupech používaných ve strojírenském podniku.

Sekundárním cílem je pak představení a demonstrace možností vývojového nástroje Wintermute Engine Development Kit pomocí kterého je výukový program vytvořen. Tato univerzální sada nástrojů umožňuje díky svojí otevřenosti vytvářet různorodé projekty. WME bude následujícím textu podrobněji představen v kapitole 3. Následně v kapitole 4 budou na projektech vytvořených pomocí tohoto nástroje předvedeny možnosti, které tato sada poskytuje.

Práce nepřímo navazuje na projekt absolventa ZČU Ing. Miroslava Suchého Manažerský simulátor MS1, který je blíže popsán v rešeršní části práce v kapitole 4.1.4. Projekt virtuální dílna je v konceptu digitální továrny a bude přispívat k usnadnění studia studentům prvních ročníků fakulty strojní a k propagaci strojírenských oborů na středních a základních školách v rámci projektu Populár.

2 Definice základních pojmů

Již v úvodu této práce bylo uvedeno několik termínů, jako virtuální realita, simulace, vážná hra. Význam těchto výrazů však nemusí být každému hned na první pohled zřejmý, nebo může být jejich chápání odlišné. Na některé dokonce existuje hned několik různých definicí. Proto si nyní uvedeme několik stěžejních termínů, které se budou vyskytovat v celém následujícím textu.

2.1 Virtuální realita

S pojmem virtuální realita si většina lidí spojuje především svět počítačových her. Virtuální realita ale začíná čím dál více nabývat na významu i v oblasti průmyslové výroby. Před několika lety sloužily virtuální počítačové modely na veletrzích jako zajímavé ukázky inovační techniky, avšak dnes se v automobilovém průmyslu stávají denní realitou. Samotný termín virtuální realita (VR) charakterizuje počítačově vygenerované prostředí, do něhož pozorovatel může zapojit své aktivity pomocí přirozených smyslů, jak by to činil v reálném prostředí [1].

Podle jiné univerzálnější definice je virtuální realita technologie, která umožňuje uživateli komunikovat s prostředím reálného nebo imaginárního světa se snahou o maximální obklopení uživatele VR, které je simulováno počítačem. Tato definice umožňuje několik různých pohledů:

- Pohled všeobjímající – takto vnímaná VR je chápaná jako umělý prostor vytvořený elektronickým zařízením. Do této definice tak spadá vše, co děláme od okamžiku zapnutí počítače až do okamžiku jeho opětovného vypnutí. Můžeme tedy připustit, že virtuální prostor je také prostředí pouličního terminálu, určeného na prodej jízdenek na tramvaj.
- Pohled obklopující – zde je za nejdůležitější aspekt považován úroveň vtažení a obklopení uživatele virtuálním prostředím. Tady již nelze tvrdit, že by uživatel při nákupu jízdenek, nebo pouhým zapnutím počítače, byl nějak vtažen do prostředí VR. Míru obklopení lze rozhodně zvýšit pomocí 3D stereoskopie. V dnešní době se většinou předpokládá, že vše, co je stereoskopické, je také virtuální realita.
- Pohled interaktivní – podle dalšího pohledu je jediná opravdová virtuální realita až v tom okamžiku, kdy si může uživatel na virtuální objekty sáhnout. Podle této definice je VR umělý prostor, v němž lze interaktivně s objekty manipulovat. Pak tedy každá i ta nejjednodušší počítačová hra spadá do oblasti virtuální reality [2].

2.2 Simulace

Simulace je napodobování procesů, nebo věcí z reálného prostředí. Samotný akt simulace obecně znamená napodobení některých klíčivých vlastností nebo chování simulovaných systémů [3].
Simulace vlastností reálného světa je stále více a více využívanější technologií. Věrné zachycení reality však zůstává velmi nákladnou záležitostí, proto je kvalita virtuálního prostředí často omezována podle výkonu hardwarových prostředků, které jsou v daném projektu k dispozici.
Simulované prostředí může na jednu stranu vypadat stejně jako reálný svět. Ukázkou mohou být například tréninkové simulace bojových situací pro výcvik pilotů. Na druhou stanu se prostředí může od reálného světa výrazně lišit. Typickým příkladem jsou zmiňované počítačové hry. V praxi je stále ještě obtížné vytvořit vysoce věrnou virtuální realitu. To je

zapříčiněno především díky technickým omezením a výkonu techniky na její tvorbu, zpracování, rozlišení obrazu a dalších technických prostředků [4].
Simulace je často používána při výcviku civilních a vojenských zaměstnanců. K tomu obvykle dochází v případě, že je příliš drahé, nebo příliš nebezpečné, aby cvičenci používali skutečné zařízení v reálném světě. V takovýchto situacích se učí nové dovednosti a získávají cenné zkušenosti v bezpečném virtuálním prostředí.
Výhodou je, že tento systém při tréninku v bezpečnostně-kritických systémech umožňuje dělat chyby bez toho, aby tyto chyby měly nějaké vážnější dopady. Jistě se všichni shodneme na tom, že je rozdíl mezi simulací používanou pro výcvik a simulací instruktážní.

Tréninková simulace obvykle spadá do jedné ze tří kategorií:

- live simulace je simulace, kde skuteční lidé využívají simulovaného zařízení v reálném světě.
- virtuální simulace je simulace, kde skuteční lidé využijí simulovaného vybavení v simulovaném světě, nebo ve virtuálním prostředí.
- konstruktivní simulace je simulace, kde lidé používají simulované vybavení v simulovaném prostředí.

Konstruktivní simulace je často označována jako válečná, protože je podobná strategickým válečným hrám, ve kterých hráči velí armádám vojáků a válečným strojům, kterými můžeme pohybovat po hrací ploše.
Simulace v oblasti vzdělávání mají určitou podobnost s výcvikovými simulacemi. Jsou však zaměřeny na konkrétní úkoly. V některých případech vytváří simulace reálné světové prostředí ve zjednodušené podobě a tím pomáhá studentovi pochopit klíčové pojmy. Uživatel pak může v rámci simulace vytvářet jakési konstrukce, které se budou chovat v souladu s předvedenou koncepcí.
Managementové hry, nebo také obchodní simulace, nachází v posledních letech oblibu v oblasti marketingu. Obchodní simulace, které zahrnují dynamický model a umožňují experimentování s obchodní strategií v bezrizikovém prostředí, poskytují užitečné diskuze o případových studiích. [3].

2.3 Vážná hra

Vážné hry jsou simulace reálných událostí nebo postupů navržených za účelem řešení nějakého problému. Primárním účelem této hry není zábava, ale vzdělávání, výcvik, výchova, nebo zdravotní péče. Přestože vážné hry mohou být i zábavné, jejich hlavním cílem je vzdělávat uživatele. V některých případech může být potlačen prvek zábavy, aby tím bylo dosaženo požadovaného pokroku ve výuce.
Nejčastější využití nalézají vážné hry v armádě pro výcvik vojáků v zacházení s bojovou technikou. V mnoha případech se k tomuto účelu využívá speciální a velmi nákladné technické vybavení. Při výcviku leteckých pilotů se například používají opravdové kokpity letadel se skutečnými ovládacími prvky, které mají místo skel instalovány velké obrazovky, kde je promítáno virtuální okolí letadla. Pilot se tak poměrně bezpečnou formou může naučit reagovat na krizové situace, které mohou nastat během plnění skutečných bojových misí [2].
S podobnými trenažéry, ne však na tak vysoké technické úrovni, se mohla setkat také většina z nás i v civilním sektoru. Například v autoškolách, kde se používají automobilové simulátory (viz Obr. 2.1), k osvojení základních dovedností budoucích řidičů před tím, než vyrazí do skutečného provozu. Na rozdíl od simulátoru hrozí ve skutečném provozu mnohem větší riziko vzniku vážných škod na majetku, nebo dokonce na zdraví ostatních účastníků silničního provozu.

Obr. 2.1 Automobilový trenažér

Pojem vážné hry byl používán dlouho před zavedením počítačů a elektronických zařízení do zábavního průmyslu. Clark Abt použil termín v roce 1970 ve své knize Serious game, publikované nakladatelstvím Viking Press. Tento pojem použil ve své knize především pro specifikaci deskových a karetních her, nicméně však zavedl obecnou definici, která je stále považována za použitelnou i v počítačovém věku: „Hra je činnost mezi dvěma nebo více nezávislými účastníky, kteří se na základě svých rozhodnutí snaží dosáhnout cílů v nějakém přesně definovaném kontextu.“ [5]

3 Popis Wintermute Engine

Vývojové prostředí Wintermute Engine Development Kit je sada nástrojů primárně určených pro vývoj počítačových her. Tento balík je primárně určen zejména na vývoj grafických point&click adventure a to jak v tradičním 2D prostředí, tak modernějším 2.5D, což je 3D postava na 2D pozadí. Sada nástrojů obsahuje interpreter her (Wintermute Engine) a grafické nástroje pro tvorbu a správu vlastního obsahu hry (WME nástroje). Dále obsahuje podrobnou dokumentaci v anglickém jazyce. Součástí vývojového balíku jsou také ukázková data a předpřipravené šablony [6]. Wintermute Engine byl navržen a naprogramován českým programátorem Janem Nedomou. První veřejná beta verze byla uvolněna 12. ledna 2003. Tento engine je stále aktivně vyvíjen a i když nepravidelně z důvodu malého vývojového týmu, je každý rok uvolněno několik aktualizací [7]. Vývojářský balík Wintermute Engine je volně ke stažení na stránkách výrobce dead-code.org v sekci download, kde jsou také uvedeny systémové nároky softwaru.

3.1 Vlastnosti enginu

Engine je založen na objektově orientované filozofii. Při vývoji softwaru jsou pro tvorbu jednotlivých objektů, jako například postav, scén, oken atd., použity jednotlivé nástroje. Každý objekt je definován svým vzhledem (grafikou, animacemi, titulky, fonty) a skripty. Tento skript definuje základní logiku chování daného objektu a jeho reakce na konkrétní události, vyvolané v průběhu programu. Objekty jsou pak v prostředí Project Manageru spojeny dohromady do jednoho celku. Project Manager zároveň zajišťuje závěrečnou kompilaci výsledného programu. Wintermute Engine obsahuje velké množství nejrůznějších funkcí a je koncipován tak, aby poskytl tvůrcům her co největší flexibilitu. Následující seznam je jen stručný výčet těch základních.

3.1.1 **Rozlišení a barvy**

Engine podporuje v podstatě jakékoliv rozlišení. Projekty mohou mít od základního rozlišení 320×200 až po Full-HD 1920×1080. Podporovány jsou barevné hloubky 16 bitů (hicolor) a 32 bitů (true color). Uživatel si může sám zvolit barevnou hloubku, odpovídající výkonu grafické karty a konverzi barev již zajistí engine automaticky.

3.1.2 **Vykreslovací subsystém**

Jelikož už jsou dnes 3D akcelerované videokarty standardem, WME dokáže využít 3D akcelerace pro rychlé vykreslování 2D grafiky ve vysokém rozlišení i s grafickými efekty, jako je průsvitnost, míchání barev a vyhlazování hran. Na starších počítačích dovede WME běžet v takzvaném „režimu kompatibility", který sice 3D akcelerátor nepožaduje, nicméně pokročilé grafické efekty jsou vypnuty.

3.1.3 **Vývojářské nástroje**

Pro zrychlení vývoje nabízí WME sadu grafických nástrojů pro návrh jednotlivých scén hry, animací a pro správu souborů hry. Nástroje interně využívají pro vykreslování přímo herní engine, takže můžeme prohlásit, že se jedná o opravdový WYSIWYG. Momentálně je ve vývoji nová generace podpůrných nástrojů.

3.1.4 **Podporované formáty souborů**

Engine umožňuje používat různé druhy souborů pro grafiku a zvuky. Kromě toho používá vlastní formáty souborů pro definování objektů. Podporované grafické formáty: BMP, TGA, PNG a JPG. WME dále podporuje PNG a TGA soubory s alfa kanálem. Podporované zvukové formáty jsou Ogg Vorbis (OGG) a WAV. Ačkoliv WME nepodporuje formát MP3 kvůli licenčním komplikacím, formát Ogg Vorbis je jeho více než rovnocennou náhradou. Tyto zvukové formáty lze použít jak pro zvukové efekty, tak pro hudbu. Velké zvukové soubory jsou během hraní "streamovány" z disku aby nezabíraly zbytečně paměť. Kromě toho umí engine přehrávat video ve formátu Ogg Theora a AVI, včetně automatického zobrazování titulků (formát SUB).

3.1.5 **Skriptovací jazyk**

WME nabízí flexibilní, objektově orientovaný skriptovací jazyk, pomocí kterého je vývojář schopen do hry začlenit téměř jakoukoliv vlastnost či hádanku. Všechny objekty nabízejí sadu metod a atributů, které umožňují snadno ovládat herní engine. Také je možné vytvářet vlastní objekty a upravovat vestavěné metody. Skriptovací jazyk používá běžnou syntaxi, podobnou jazyku C, podobně jako JavaScript, C++, C#, Java či PHP [6]. Díky čemuž je poměrně snadné se rychle naučit psát skripty pro WME.

3.1.6 **Paralaxní skrolování**

Paralaxní skrolování je jedna z technik v počítačové grafice, která simuluje 3D hloubku pohyblivého obrazu ve 2D. Tato technika se používá již od 40. let 20. století, kdy našla uplatnění zejména v animovaném filmu. Další rozšíření nastalo počátkem 80. let 20. století v oblasti herního počítačového průmyslu. Slovo „paralaxa" vzniklo spojením slov „parallel axis" neboli „rovnoběžné osy", slovo „skrolování" znamená posun [8]. Technika paralaxní skrolování spočívá v rozdělení scény do několika vrstev. Každá z těchto vrstev se na scéně posouvá jinou rychlostí, což budí dojem prostorové perspektivy. Tato technika vícevrstvé scény je přímo podporována enginem a nástrojem pro návrh scén, proto není třeba pro paralaxní skrolování psát dodatečný skript.

3.1.7 **Balíkovací systém**

Než je hotový produkt distribuován, může být zabalen do jednoho nebo více "balíků". Balík obsahuje všechny soubory software ve zkomprimovaném tvaru. Můžete si zvolit, jestli chcete program rozdělit do více balíků či nikoliv. Jednotlivé balíky je pak možné nabízet zvlášť v případě, že chcete výsledný produkt poskytovat po jednotlivých kapitolách, nebo zvlášť program a zvlášť zvuky. Tyto balíky mohou mít různou prioritu. Toho lze využít třeba při vydávání opravných patchů. Balík - patch bude obsahovat pouze změněné soubory s vyšší prioritou, než původní balíky. Engine pak použije přednostně změněné soubory. Během vývoje není třeba tyto balíky kompilovat. Engine dokáže pracovat přímo nad zdrojovými soubory, což ulehčuje vývoj i ladění.

3.1.8 **Vrstva pro tvorbu uživatelského rozhraní**

Pomocí několika základních stavebních prvků (jako jsou okna, tlačítka atd.) lze sestavit komplexní uživatelské rozhraní. Například okno pro nahrání/uložení, okno nastavení, inventář a podobně. Všechny ovládací prvky jsou plně „skinovatelné", díky čemuž lze kompletně změnit jejich vzhled.

3.1.9 **Podpora lokalizace**

WME umožňuje přeložit software do jiných jazyků. Lokalizace není omezena pouze na texty, ale lze vytvořit lokalizační balíky pro jednotlivé jazyky. Tyto balíky mohou obsahovat přeloženou tabulku textů, písma, grafiku nebo dokonce dabing v jiném jazyce.

3.1.10 **Podpora a usnadnění**

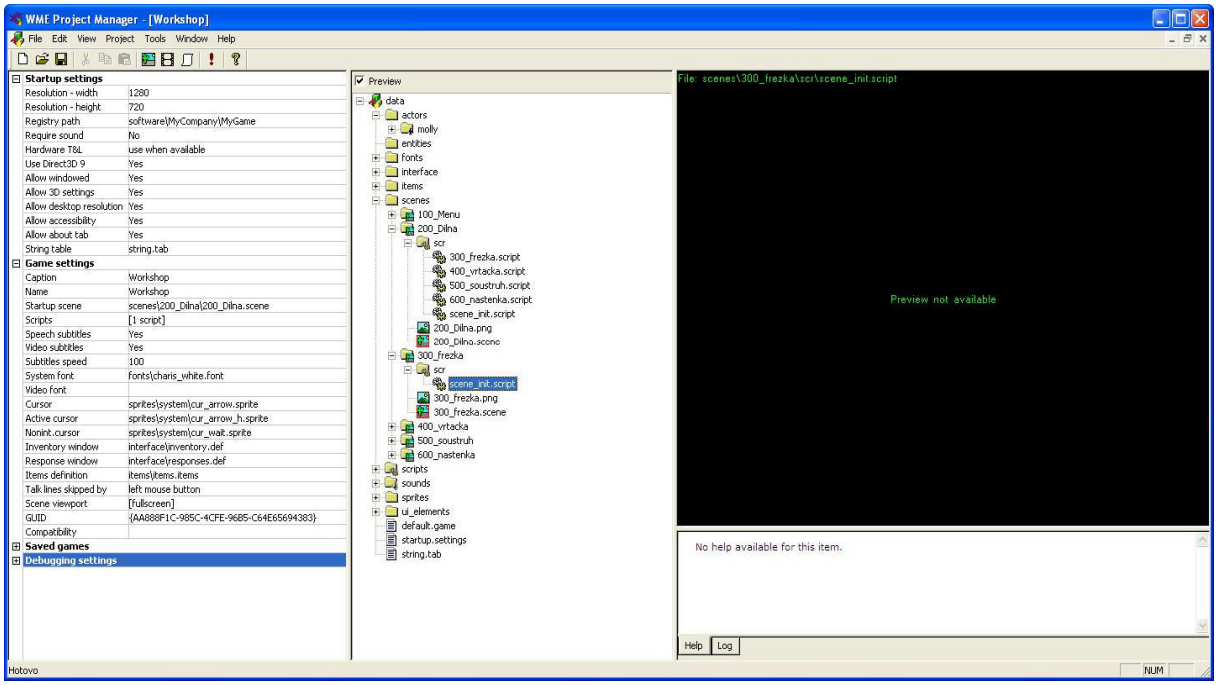
WME nabízí několik možností usnadnění pro zrakově postižené uživatele. Jednak má možnost zapnout přesměrování psaného textu do hlasového syntezátoru. Dále je možné pomocí klávesové zkratky zvýraznit aktivní místa na obrazovce, jejichž nalezení může být pro zrakově postiženého leckdy obtížné. Poslední možností je možnost kdykoliv pozastavit hru pomocí klávesové kombinace Ctrl+Mezerník [6].

3.2 Struktura programu Wintermute Engine

Balík Wintermute Engine je tvořen několika relativně nezávislými moduly, které jsou podle potřeby volány z hlavního okna programu. Každý modul je spouštěn ve svém vlastním okně a je schopen běžet nezávisle na hlavním okně. Výhoda tohoto modelu je možnost současné editace několika vlastností najednou a volného přepínání mezi jednotlivými moduly, což značně urychluje práci s programem. Pro příklad můžeme uvést situaci, kdy je při editaci skriptu nutné zjistit konkrétní souřadnice na scéně. Okno se skriptem lze jednoduše minimalizovat do lišty a v novém okně otevřít editor scén, kde provedeme naměření souřadnic, aniž bychom museli přerušit práci na psaní skriptu. V následujícím textu si podrobněji představíme jednotlivé moduly.

3.2.1 **Project Manager**

Hlavní okno programu je nazváno Project Manager a tvoří základ celého vývojářského balíku. Z tohoto okna se spouštějí hlavní funkce programu a také odtud probíhá volání ostatních modulů. Okno Project Manageru je rozděleno do čtyř hlavních částí, jak je patrno z Obr. 3.1.



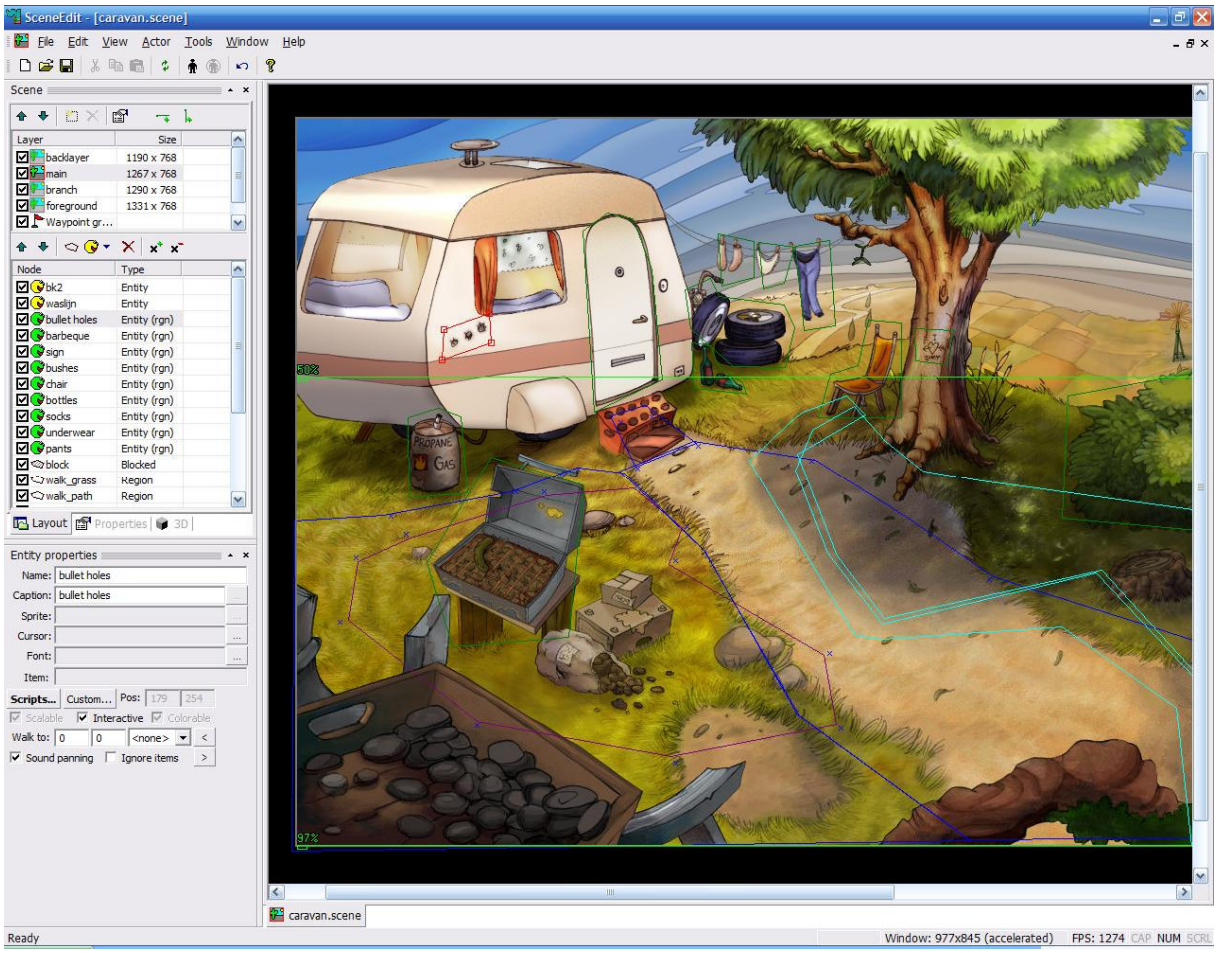
Obr. 3.1 Hlavní okno WME

Dále obsahuje v horní části okna nástrojovou lištu, pomocí které lze spouštět další funkce, nebo jednotlivé moduly.
V levé části okna je umístěn panel s parametry pro výchozí nastavení hry, který je dále rozčleněn do čtyř skupin podle svého účelu. Ve skupině Startup setting se nachází základní systémové nastavení, jako je výchozí rozlišení obrazovky, způsob použití hadrwarové akcelerace grafického prostředí, místo pro umístění záznamů o nastavení programu v registrech Windows a další. Skupina Game settings obsahuje základní informace a možnosti nastavení programu. Jsou zde umístěny parametry typu Caption, odkaz na startovací obrazovku, adresa hlavního skriptu programu, výchozí doba zobrazování textu, jaký font a jaký kurzor má být použit a další vlastnosti. Třetí skupina obsahuje možnosti nastavení pro ukládání pozic a čtvrtá parametry pro nastavení nástroje pro ladění a vyhledávání chyb.
V prostředním panelu je umístěn souborový průzkumník. Průzkumník má strukturu grafického stromu, díky čemuž má vývojář dobrý přehled o tom, v jaké části adresářové struktury vyvíjeného produktu se právě nachází.
Struktura projektu je rozčleněna do složek, podle typu objektu. Pro příklad složka scenes obsahuje další podsložky, v nichž jsou uložena data týkající se vždy jedné konkrétní scény. Jsou zde obrázky scény, skripty a sprites týkající se dané scény a samozřejmě také soubor scény, kde jsou uloženy logické vazby výše uvedených objektů.
Tento průzkumník má také integrovanou funkci náhledů, která se povoluje v horní části panelu. Ihned po najetí myší na daný soubor má vývojář možnost vidět náhled obrázku, nebo vzhled vybraného fontu. Tato funkce umožňuje dokonce náhled podporovaných zvukových souborů, aniž by bylo nutné je otevírat v externím přehrávači. Náhledy bohužel nejsou funkční při zobrazení samotného kódu skriptu a scén, proto je tyto soubory vždy nutné otevřít v modulu, k tomu určenému.
Po kliknutí pravým tlačítkem myši na konkrétní složku nebo soubor, se zobrazí nabídka s dalšími funkcemi. Základními funkcemi, jako jsou Explore, Create subfolder, Remove a Refresh, asi nemá smysl se zde příliš zabývat, avšak o to zajímavější jsou funkce, které přímo souvisí s vývojem programů. Pomocí funkce Add lze vytvářet přímo do zvoleného místa adresářového stromu nové objekty, jako jsou entity, okna, scény, skripty a to včetně předdefinované základní struktury souboru. Funkcí import lze zkopírovat do adresářové struktury projektu jakýkoliv již existující soubor nebo složku z jakéhokoliv místa na disku počítače.
V pravé části okna Project manageru je umístěna oblast pro zobrazování náhledu grafického obsahu jednotlivých souborů, včetně animovaných sprite a náhledů oken. Náhledy jsou generovány přímo enginem, proto jejich zobrazení je věrné a vzhled je přesně takový, jaký bude při běhu software. Pod grafickou oblastí se nachází dvě záložky, umožňující zobrazení krátké nápovědy nebo zápis chyb z běhu programu, uložených v logovacím souboru.

3.2.2 **Editor Scén**

Dalším modulem, se kterým přijde vývojář do styku, je editor scén. V tomto modulu se editují jednotlivé lokace, včetně umisťování aktivních oblastí a dalších entit. V horní části okna editoru je nástrojová lišta, nabízející mimo běžných funkčních tlačítek také možnost umístit na scénu postavy předpřipravené v modulu Sprite Editor. Modulem pro editaci postav se budeme zabývat v následující kapitole. Kromě uvedeného tlačítka lze postavu také přidat přímo ze skriptu k tomu určeným příkazem.
Levá část okna editoru je rozdělena na dvě oblasti. V horní polovině této oblasti je umístěn seznam entit, které jsou použity ve scéně. Tyto entity mohou být řazeny do různých skupin, které lze individuálně konfigurovat. Výčet těchto skupin jsou první položky seznamu, jak je patrno na Obr. 3.2.

Pod oblastí znázorňující skupiny se již nachází výčet jednotlivých entit. Entity jsou rozlišeny ikonami podle svého typu. Zde lze také určit, zda má být entita ve výchozím stavu aktivní či nikoliv. Tento parametr lze později v průběhu programu přepínat na základě nějaké události přímo ze skriptu.



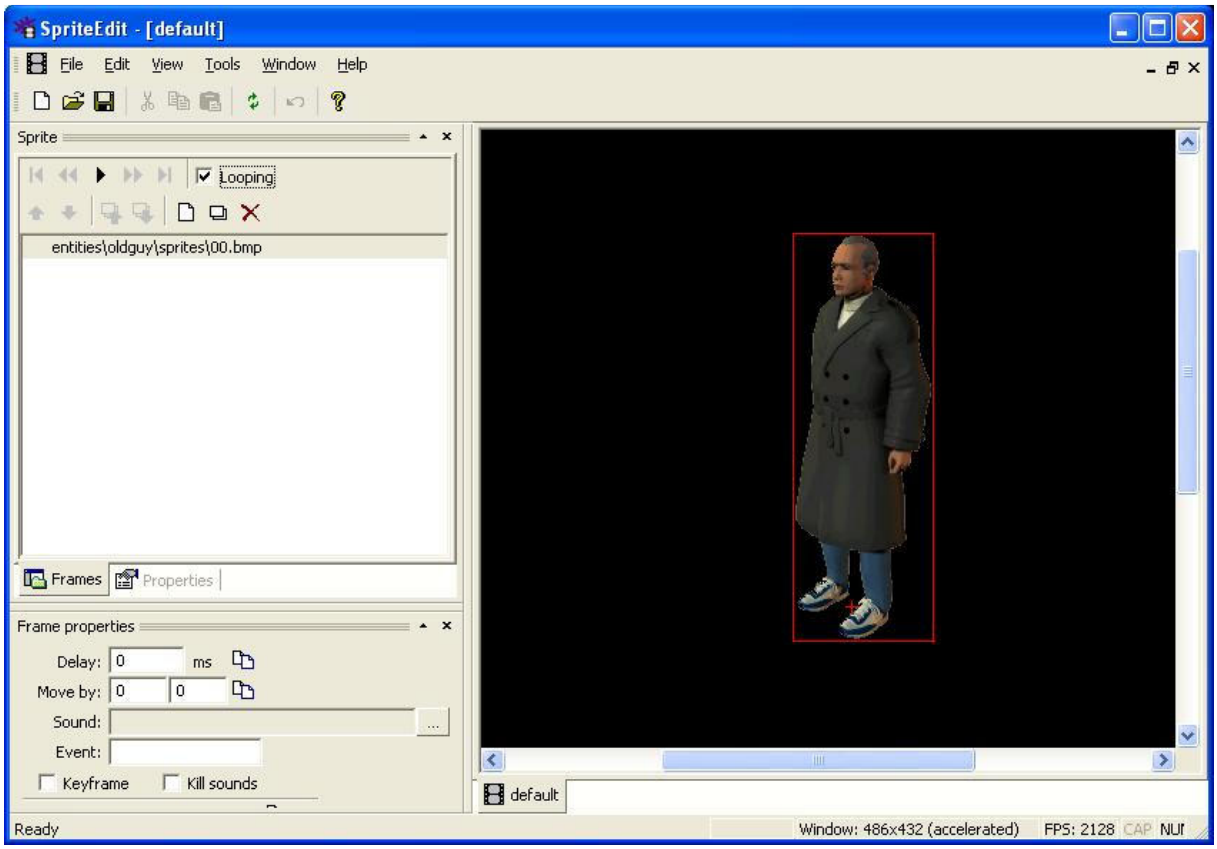
Obr. 3.2 Editor scén

Každá z vložených entit má svoje vlastnosti a připojené skripty. Tyto vlastnosti lze konfigurovat ve spodní polovině této oblasti. Zde lze také entitě také buď přiřadit existující skript, nebo vytvořit nový a rovnou jej otevřít v modulu určenému pro jeho editaci.
Pravá část okna editoru scén je vyhrazena pro zobrazování náhledu budoucí scény. Kromě pozadí scény jsou zde vyobrazeny také jednotlivé entity typu sprite, což jsou objekty většinou v podobě dvourozměrného obrázku, který se může na základě nějaké události na scéně objevit, zmizet, nebo se po ní pohybovat. Dále jsou zde barevnými rámečky označeny aktivní oblasti tzv. regiony a zelené čáry měřítka. Tato měřítka určují, jakým způsobem se bude měnit velikost postavy během pohybu po scéně. Všechny tyto objekty lze v oblasti náhledu volně přesouvat, měnit jejich tvar a dále specifikovat jednoduše pomocí myši.

3.2.3 **Sprite Editor**

Pohyblivé i všechny ostatní objekty integrované do scény lze vytvářet pomocí Sprite Editoru, který je dalším modulem balíku WME. V jeho okně nalezneme nástrojovou lištu umístěnou v horní části okna. Hlavní část tohoto modulu je rozdělena na tři části. V pravé části je umístěna oblast umožňující náhled vytvářeného objektu podobně, jako tomu bylo u předchozího modulu.

Levá část okna je pak rozdělena na dvě oblasti. V horní polovině můžeme vytvářet pohyb editovaného objektu pomocí sekvence po sobě jdoucích obrázků. Zde je nezbytné, aby jednotlivé obrázky měly správně definované souřadnice určující jeho polohu na scéně. V opačném případě by výsledná animace nebyla plynulá. Obrázky animací lze také vkládat do hierarchicky řazených sestav, což možnosti animací značně rozšiřuje. Výslednou animaci lze okamžitě přehrát v okně náhledu přímo ze Sprite Editoru pomocí tlačítek umístěných v horní části editačního okna. Ovládání těchto tlačítek je zcela intuitivní. Vedle těchto tlačítek můžeme povolit funkci Looping, která nám zajistí nepřetržité opakování animace a tím usnadní kontrolu a dohledávání případných nepřesností pohybu.
Pod touto editační částí je v levé spodní části okna umístěn panel, který umožňuje změnu vlastností každého snímku vytvářené animace nezávisle na ostatních. Lze zde například nastavit dobu zobrazení jednotlivého snímku a posun snímku po scéně oproti předchozímu. Tím lze vytvořit pohyb objektů po scéně. Také se zde mohou přiřadit soubory s melodiemi, nebo zvuky, které se mají přehrát během animace, nebo lze zde potlačit přehrávání doprovodné hudby. Celkový vzhled sprite editoru je patrný z následujícího obrázku (Obr. 3.3).

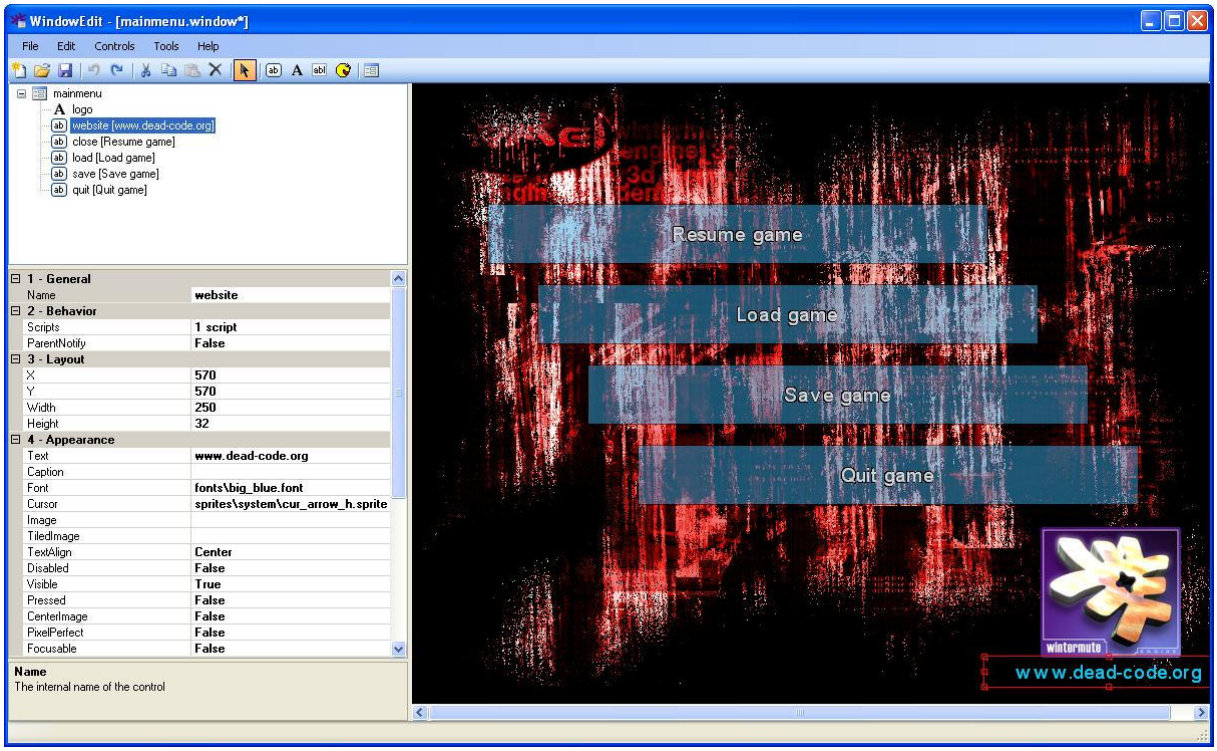


Obr. 3.3 Sprite editor

Kromě tvorby pohyblivých objektů umožňuje sprite editor vytváření také objektů nepohyblivých. Tyto objekty jsou využívány zejména tehdy, když je potřeba ovlivnit jejich zobrazování na scéně. Takto vytvořený objekt – sprite, můžeme snadno v průběhu programu odstranit ze scény příkazem přímo ze skriptu. Princip vytváření takového nepohyblivého objektu je totožný s výše uvedeným postupem, jen postačí vložit pouze jeden obrázek do stromu animací.

3.2.4 Editor Oken

Jednou z velkých předností WME je způsob, jakým je umožněno vytvářet okna vyvíjené aplikace. K tomuto účelu slouží modul Editor Oken, který je další součástí tohoto enginu. Modul poskytuje vývojáři maximální volnost při editaci.
Okno editoru je zachyceno na obrázku Obr. 3.4. Jak je z obrázku patrno, okno má opět ve své horní části nástrojovou lištu, která nabízí kromě základních funkcí také tlačítka, umožňující přidávání nových objektů. Pomocí těchto tlačítek lze přidávat prvky, jako jsou tlačítka, popisové pole, textové pole, kontejner entit, nebo přidávat další podokna, která je možno dále hierarchicky řadit.
Seznam všech objektů v okně se nachází v levé horní části okna editoru, kde je možné tyto objekty volně přesouvat myší nahoru a dolů, nebo je přemístit do případných dalších podoken.
Pod tímto seznamem se nachází soupis všech volitelných parametrů zvoleného objektu. Kromě základních parametrů, jakými jsou například rozměry a umístění objektu, nastavení viditelnosti objektu, vložení podkladového obrázku objektu a dalších parametrů, lze zde také ke každému objektu přiřadit skript a tak dále rozšiřovat jeho funkcionalitu.

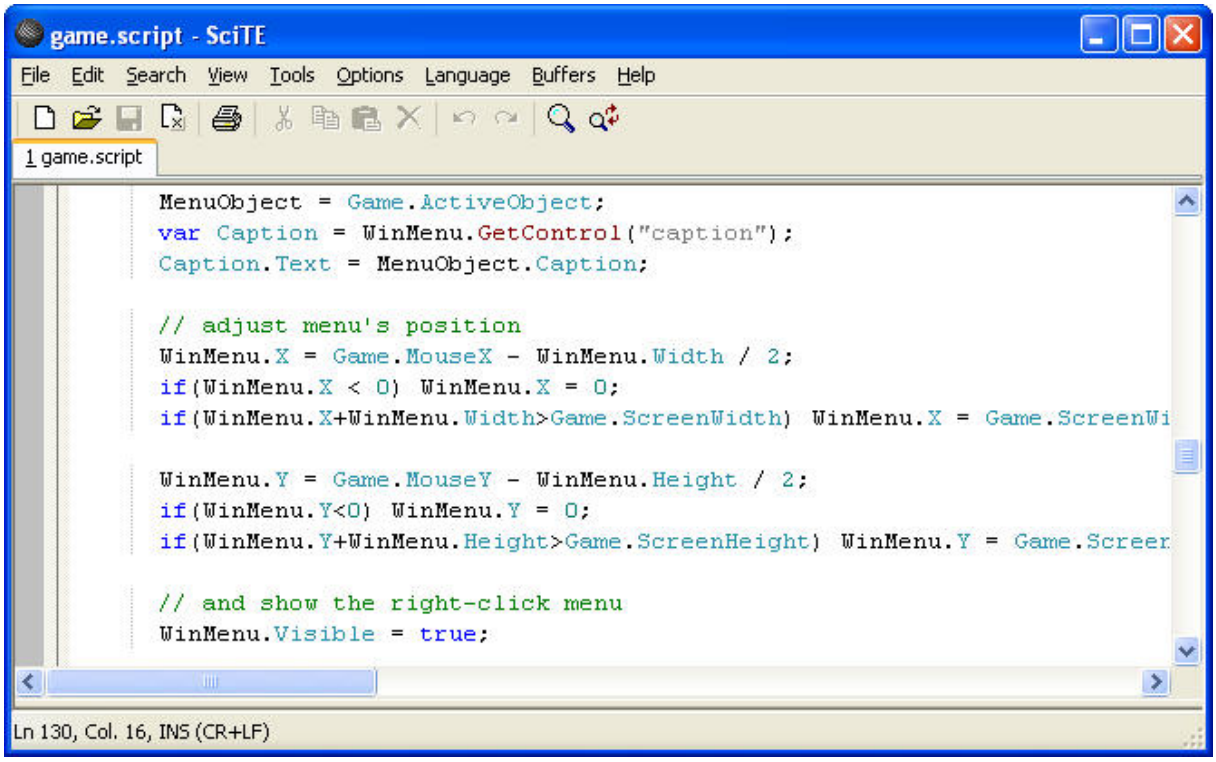


Obr. 3.4 Editor oken

Pod prostorem s výše uvedenými parametry je umístěna krátká nápověda k právě zvolené položce editoru.
Pravá část okna editoru je, stejně jak tomu bylo u všech předchozích modulů balíku, vyhrazena pro zobrazování náhledu. Při označení určitého objektu kurzorem se zde zvýrazní jeho aktivní oblast, které lze uchopením za její klíčové body upravovat, nebo přesouvat kamkoliv v rámci navrhovaného okna.
Výsledné návrhy, které vzejdou z této komponenty, mohou, díky širokým možnostem které modul poskytuje, vypadat velice efektně a vůbec nemusí mít podobu klasického okna.

3.2.5 Editor skriptů

Tento editor není produktem tvůrce enginu, ale jedná se o tzv. software třetí strany. Za účelem vytváření skriptů byl do vývojářského balíku přidán textový editor SciTE založený na knihovně Scintilla. Scintilla je komponenta, sloužící pro volnou úpravu zdrojového kódu. Knihovna je dodávána s kompletním zdrojovým kódem a je volně šířena včetně zdrojových kódů pod licencí MIT, která umožňuje použití tento produkt v jakémkoliv volném nebo proprietárním produktu pod podmínkou, že text licence MIT bude dodán společně se softwarem. Programy šířené pod touto licenci lze také kombinovat s jinými produkty, které jsou šířeny bod licencí GPL, protože licence GPL tuto kombinaci výslovně povoluje. Další software šířený pod touto licencí je například SSH klient PuTTY, skriptovací jazyk Lua, nebo sada nástrojů Mono [9].
Stejně jako ostatní standardní komponenty pro úpravy textu i Scintilla obsahuje funkce, užitečné zejména při editaci a ladění zdrojového kódu. Mezi hlavní patří zejména zvýraznění správné syntaxe, možnost sbalit editovaný text do bloků, indikátor překlepů, doplňování kódu a zobrazování nápovědy (Hint). Knihovna dokáže také označit místo a řádek, kde došlo během ladění k přerušení běhu programu. Flexibilita Scintilly je oproti podobným editorům mnohem větší. Mimo jiné umožňuje použití proporcionálního písma, tučného a kurzívního písma, různé barvy pozadí a použití více fontů.



Obr. 3.5 Editor skriptů SciTE

SciTE (Obr. 3.5) je multiplatformní textový editor založení na výše uvedené komponentě Scintilla. První verze tohoto editoru vyšla v květnu 1999. Původně byl určený pouze k demonstraci možností Scintilly a aby prokázal, že je plnohodnotná a použitelná komponenta vhodná pro vytváření a spouštění programů. SciTe je lehký a velmi rychlý, nejen proto se nejvíce hodí pro práci s jednoduchými aplikacemi a skripty [9]. Přestože SciTE neobsahuje mnoho možností nastavení v konfiguračních oknech, jedná se o velmi univerzální a vysoce konfigurovatelný editor. Nastavení programu se provádí přímou úpravou textových konfiguračních souborů. Je možné zadat pro každý programovací jazyk individuální nastavení, nebo globální nastavení pro konkrétního uživatele editoru. V současné době je

SciTE lokalizována do více než dvaceti jazyků a podporuje více než 36 programovacích jazyků [10]. Scintilla a SciTE jsou k dispozici pro operační systémy Windows XP a novější, OS X a GNU/Linuxové systémy založené na knihovně GTK + [9].
Editor skriptů se vyznačuje především svojí jednoduchostí a přehledností. Ve standardním nastavení je zobrazována v horní části okna jednoduchá nástrojová lišta s běžnými funkcemi, pod kterou se nachází lišta záložek. Pomocí této lišty lze přepínat mezi jednotlivými otevřenými skripty, což přispívá k rychlejší práci. Ve střední části okna se již nachází oblast pro zapisování programového kódu. Pod touto oblastí je umístěn stavový řádek, který obsahuje informaci o horizontální a vertikální pozici kurzoru, režimu psaní textu (vkládání nebo přepis) a způsobu zakončování řádky.

4 Analýza možností enginu Wintermute Enginu

Jak již bylo předesláno v předcházející kapitole, WME je velmi flexibilní nástroj, s nepřeberným množstvím funkcí. Pomocí tohoto enginu lze vytvářet počítačové hry i programy se vzdělávacím obsahem. Vytvořené programy jsou plně soběstačné a po dodržení určitých zásad se dají spouštět na všech běžných platformách. Vytvořené aplikace mohou libovolně kombinovat grafické, zvukové i textové objekty do jednoho celku a tak vytvářet velice efektní produkt.

4.1 Projekty vytvořené ve WME

Tento univerzální nástroj si oblíbilo mnoho vývojářů po celém světě, proto můžeme říci, že WME přispělo ke vzniku velkého množství softwaru. Nyní představením několika z nich budou předvedeny možnosti, které balík Wintermute Engine poskytuje.

4.1.1 **Alpha Polaris**

Děj této hry se odehrává uprostřed sněhových plání Grónska, kde se nachází stejnojmenná Americká ropná výzkumná stanice. Vysoko nad ní se právě formuje nejsilnější iontová bouře za posledních sto let, která ovlivňuje vnímání reality výzkumných pracovníků a přináší zvláštní směsici reality a nočních můr [11]. Jedná se o klasickou point & click adventuru s dobře propracovaným systémem dialogových oken. Jak je patrno z obrázku (Obr. 4.1) hra obsahuje graficky povedené scény s velmi členitým prostředím.

Obr. 4.1 Alpha polaris

Jsou zde naplno využity možnosti editoru oken WME. Okna jako inventář, nebo hlavní menu jsou přepracovány k nepoznání. Dokonce i rozhovory jsou zde řešeny pomocí speciálně upravených oken.

Dalším výrazným prvkem tohoto produktu jsou tzv. minihry, které jsou vhodně integrovány do celého děje. V rámci těchto miniher hlavní postava řeší další úkoly, jako například stopování uprchlého psa pomocí GPS vysílače, připevněného k jeho obojku. Ke stopování postava využívá notebook, který je opět řešen jako speciálně upravené okno.

4.1.2 **Kulivočko**

Tato hra je dobrodružným příběhem malého chlapce jménem Kulivočko, který opustí zdi sirotčince, aby nalezl své kořeny a odhalil minulost svých předků. Později se mu však zjeví kouzelný dědeček a poprosí ho, aby se vydal hledat tajemný talisman a tím zachránil pohádkovou zem [11].

Obr. 4.2 Kulivočko

Hra je cílena především na mladší hráče a jednoznačně se v ní nezapřou určité výchovné prvky, které uživatel postupně nevědomky vstřebává a ty pak působí na utváření jeho charakteru. Tato hra se pokouší uživatele zaujmout pohádkovou grafikou, poutavou hudbou, která dodává celé hře tu správnou atmosféru, promyšleným dějem a mnoha důmyslnými úkoly. Je zde také mnoho logických hádanek a nejrůznějších zámků, které prezentují možnosti enginu vytvářet jednotlivé, vzájemně provázané události do různých celků, kde jedna činnost ovlivní způsob, kam se bude dále ubírat vývoj celého příběhu.

4.1.3 **J.U.L.I.A.**

Další produkt vytvořený ve WME, který bude nyní představen je futuristická sci-fi hra J.U.L.I.A. Tato hra je koncepčně rozdělena na dvě části. První část se odehrává na palubě kosmické sondy, která je ovládána umělou inteligencí. Podle jména této umělé inteligence byla také pojmenována celá hra. Jako jediný přeživší člen posádky zde uživatel prozkoumává okolní planety, opravuje a provádí údržbu sondy a cestuje mezi planetami aktuální sluneční soustavy. Druhá část hry je zaměřena na těžbu materiálu, zkoumání opuštěných

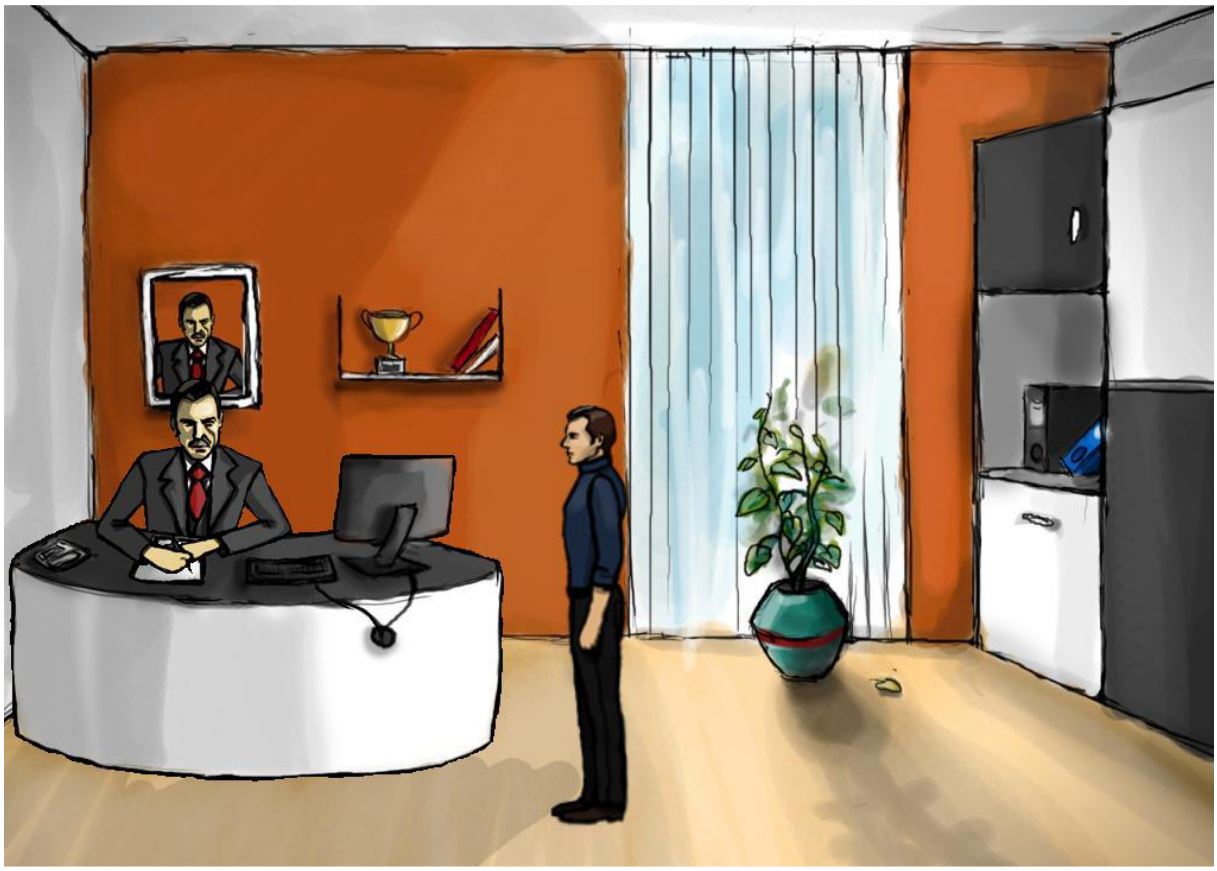
základen na planetách a na komunikaci s tamními domorodci [12]. Hra je velmi podařená po grafické stránce, což je patrno na následujícím obrázku (Obr. 4.3).

Obr. 4.3 J.U.L.I.A

Tento produkt brněnského studia CBE Software je opravdu na velmi vysoké úrovni a to jak po grafické a zvukové tak i po dějové stránce. Je obdivuhodné, čeho byli programátoři jmenovaného studia s balíkem WME schopni dosáhnout, a to i přes veliké finanční potíže, se kterými se projekt během celého vývoje potýkal.
Jak je patrno z materiálů na stránkách výrobce, výchozí šablony všech objektů obsažených ve Wintermute Engine byly přetvořeny k nepoznání. Interaktivní dialogová okna, jednotlivá submenu, která se zobrazují při plnění dílčích úkolů, okno výzkumné laboratoře, kde je možno provádět analýzu nalezených předmětů. To je jen stručný výčet základních prvků, které dokazují, že WME je velice výkonný a všestranný nástroj, umožňující tvorbu od jednoduchých dětských her, až po rozsáhlé projekty, jakým J.U.L.I.A. zcela jistě je.

4.1.4 **Manažerský simulátor MS1**

Program MS1 je výtvorem absolventa ZČU Ing. Miroslava Suchého. Jedná se o výukový program, který se na rozdíl od výše uvedených produktů nezaměřuje tolik na to, aby ohromil uživatele propracovanou 3D grafikou, strhujícím hudebním doprovodem a emotivním příběhem. Hlavní důraz je zde kladen na obsahovou část programu a na jeho edukativní přidanou hodnotu.
Dle slov autora je program primárně určen jako pomůcka při studiu vybraných metod průmyslového inženýrství zábavnou formou. Hlavní postava jménem Greg zastává nižší vedoucí pozici ve fiktivní firmě. Z této pozice plní vložené úkoly, které souvisí s řízením podniku. Touto formou jsou studentovi názorně vysvětleny základní principy logistických metod Kanban, Just in time a 5WHYS [13].
Manažerský simulátor je rozdělen do 7 různých lokací, ve kterých se celý děj odehrává. V kanceláři ředitele, kterou zachycuje obrázek (Obr. 4.4), dostává uživatel jednotlivé úkoly a také zde tyto úkoly zakončuje.



Obr. 4.4 Manažerský simulátor MS1

Kancelář sekretářky slouží jako místo, kde se student může dozvědět doplňující informace o probírané látce a také je zde poprvé upozorněn na skutečnost, že má k dispozici informační tablet, který obsahuje podrobné informace o jednotlivých studovaných metodách.
Dílna je hlavním místem, kde student řeší zadané úkoly. Tady komunikuje s virtuálním zaměstnancem a probíhá zde praktická ukázka logistických metod.
Ve skladu uživatel komunikuje s další virtuální postavou, a sice postavou skladníka. Jsou mu zde předvedeny praktické důsledky nedodržování správného postupu při provozování metody Kanban a může si zde během rozhovoru se skladníkem vyzkoušet metodu 5WHYS. Ve skutečnosti je sklad rozdělen na 3 různé lokace, které se přepínají podle situace vyplývající z běhu děje.
Poslední lokací je chodba, ve které celý simulátor začíná a zároveň slouží jako logické propojení mezi jednotlivými scénami.
Nejdůležitější edukativní pomůckou v tomto projektu je zmiňovaný informační tablet s popisem principu jmenovaných logistických metod, který má student neustále k dispozici. Zde se může dočíst potřebné informace kdykoliv nebude něčemu rozumět. Nově nabyté vědomosti bezprostředně aplikuje v simulovaném prostředí virtuálního podniku. Tablet je vytvořen pomocí sestavy velkého množství oken, které se postupně otevírají jedno na druhé tak, že již nastudovaný text je překryt textem novým. Samotný studijní obsah byl vytvořen v textovém editoru, odkud byl pomocí klávesy PrintScreen ofocen a převeden do obrázku. V této formě je vkládán do okna tabletu.
Kromě podkladového obrázku se studijním textem obsahuje každé okno několik navigačních tlačítek, které umožňují změnu zobrazovaného textu. Pro zavření tabletu je připraveno tlačítko, které je umístěno na logu ve spodní části tabletu. Po kliknutí na toto tlačítko se tablet zavře.

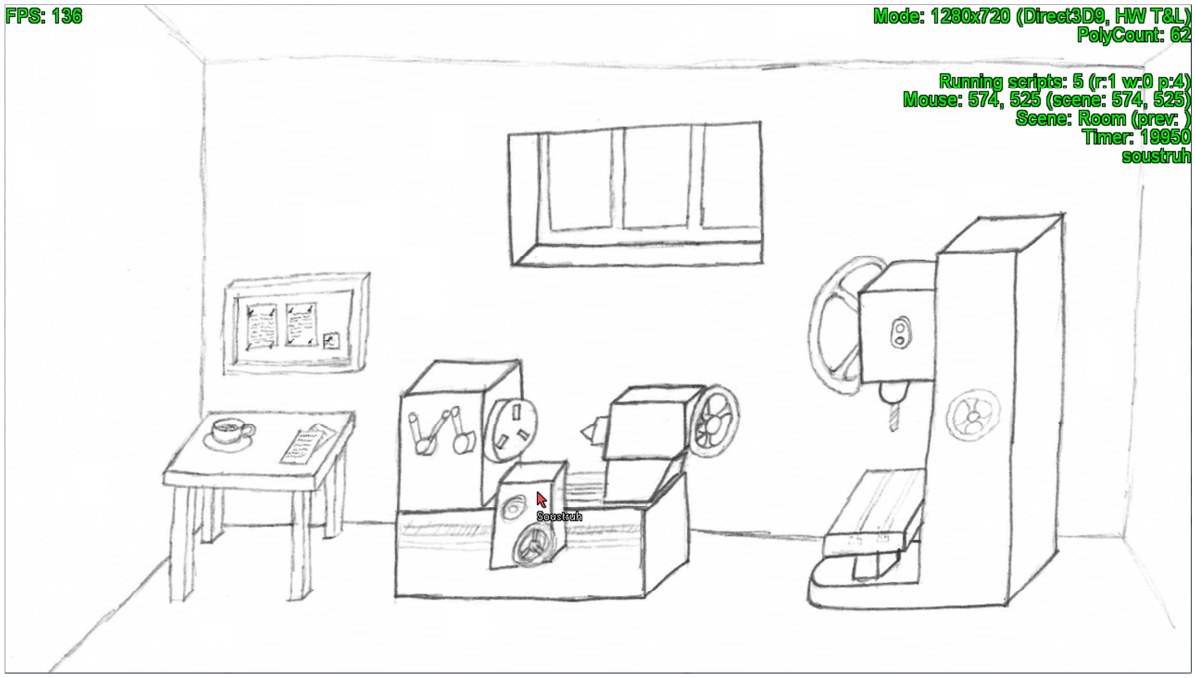
Takto zvolený systém oken však způsobuje, že pokud uživatel v tabletu studuje delší dobu a postupně přepíná mezi jednotlivými listy, nahromadí se pod aktuálním oknem velké množství starých oken a při zavření tabletu dojde k postupnému uzavírání všech otevřených oken. Tento proces je sice zautomatizován, ale bohužel je patrný pouhým okem a působí poněkud rušivým dojmem.

5 Tvorba programu

Jak již bylo předesláno v úvodu práce, hlavním cílem projektu je tvorba výukového programu v konceptu digitální továrny. Výukový program bude sloužit především těm studentům, kteří dosud neměli ve svém profesním životě příležitost setkat se s představovanými obráběcími stroji a umožní jim získat základní představu o jejich fungování.

5.1 Výběr vhodného rozlišení

Ve snaze zajistit kompatibilitu s co možná největším počtem počítačů byl v první fázi projektu vytvořen z původní skici podoby dílny (viz Obr. 5.1) jednoduchý program v několika variantách s různým grafickým rozlišením. Tyto varianty byly následně podrobeny testu na několika počítačích s různou konfigurací, aby byla nakonec vybrána jedna varianta s optimálními parametry.



Obr. 5.1 Skica dílny

Během testování nastaly největší problémy ohledně zobrazovacího zařízení připojeného k počítači. V případě, že monitor nepodporuje rozlišení, ve kterém byl software vytvořen, je zobrazena pouze černá obrazovka bez možnosti provádět jakékoliv akce, nebo je aplikace ukončena s chybou.

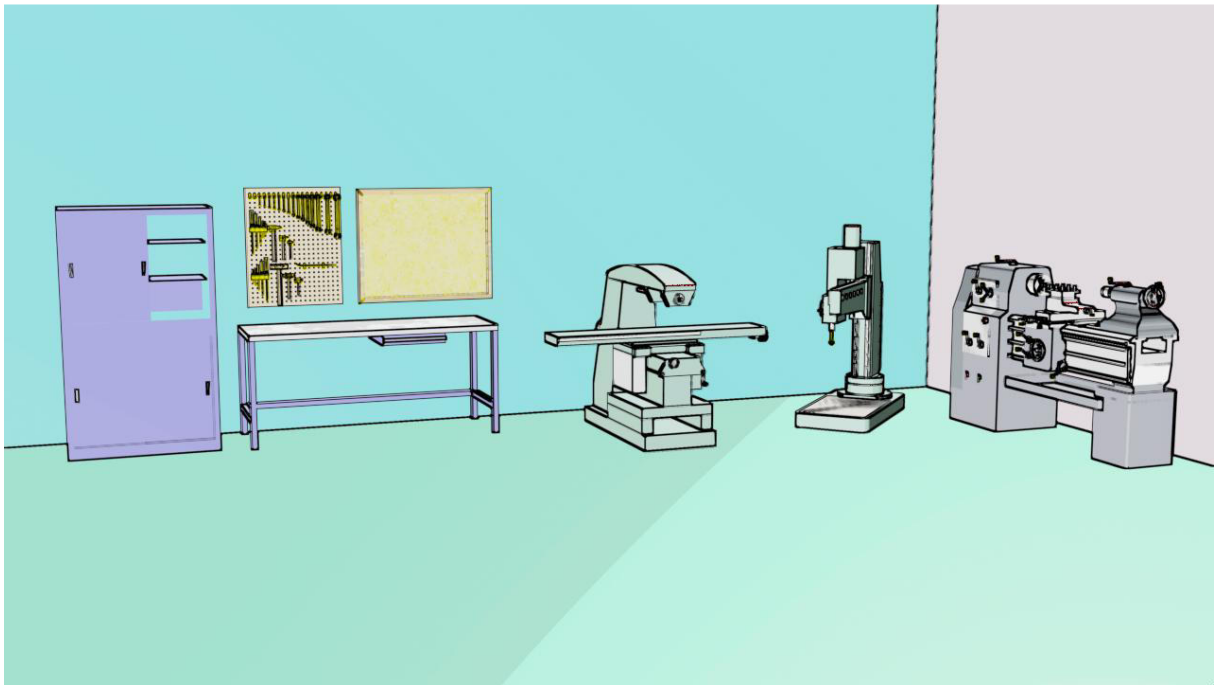
	Technologie monitoru	Rozlišení monitoru					
		1280x720		**1366x768**		**1920x1080**	
		full-screen	okno	full-screen	okno	full-screen	okno
Acer V193	LCD (4:3)	v boxu	funkční	černá obr.	funkční	černá obr.	funkční
AOC 9K+	CRT (4:3)	v boxu	funkční	v boxu	funkční	pád aplikace	funkční
LG M2432	LCD (16:9)	funkční	funkční	funkční	funkční	funkční	funkční
Notebook HP	LCD (16:9)	funkční	funkční	funkční	funkční	černá obr.	funkční
Mininotebook Amilo Mini	LCD (16:9)	pád aplikace	funkční	pád aplikace	funkční	pád aplikace	funkční

Tab. 5.1 Výsledky testování monitorů

Výběr správného rozlišení, ve kterém bude aplikace vytvořena, je proto velmi důležitým rozhodnutím. Výsledky zmiňovaného testovaní, které proběhlo v rozlišení 1920x1080, 1366x768 a 1280x720 jsou patrné z Tab. 5.1.
Z uvedených výsledků vyplývá, že největší pravděpodobnost úspěšného zobrazení je při rozlišení 1280x720. Dá se předpokládat, že v současné době bude toto rozlišení podporovat téměř každý monitor. V případě, že by toto rozlišení přece jen podporováno nebylo, stále zde bude ještě možnost spustit aplikaci v okně. Standardní okno systému Windows však působí poněkud rušivě, proto bude lépe se této variantě pokud možno vyvarovat.

5.2 Grafický interface

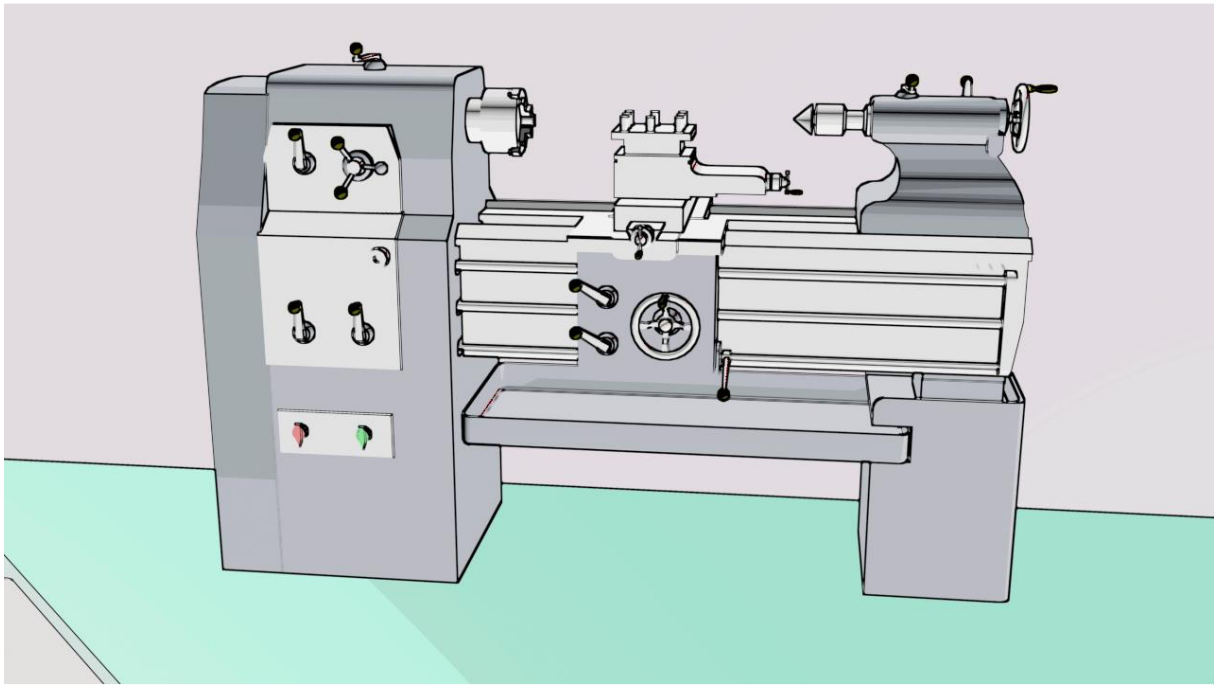
Po stanovení optimálního rozlišení aplikace, vyvstal další neméně důležitý úkol a to vybrat odpovídající grafickou podobu projektu. Po vzhledové stránce by měl být program jednak líbivý, což zvýší atraktivitu programu pro koncové uživatele, ale zároveň musí být, s přihlédnutím k zamýšlenému účelu aplikace, zachována určitá reálnost vytvořeného prostředí. Po zvážení několika variant byl vybrán model, kdy je prostředí simulované dílny rozděleno do několika pohledů (lokací), které zobrazují jednotlivé stroje. Tyto lokace jsou vzájemně spojeny jedním pohledem na celou dílnu. Program je dále rozdělen do dvou režimů, režimu studijního a režimu praktického. Studijní režim plní hlavní účel tohoto programu. V rámci tohoto režimu si student může prohlížet dílnu a seznamovat se se základními technickými údaji o jednotlivých strojích a popisem jejich hlavních částí. Zároveň má kdykoliv k dispozici ucelenější studijní text ve formě virtuálních skript, kde má možnost si prostudovat detailnější informace o problematice daného stroje. V rámci praktického výrobního režimu má student možnost si nabyté znalosti ihned ověřit v simulované praxi.



Obr. 5.2 Prostředí virtuální dílny

Z důvodu zachování možnosti pokračování ve vývoji programu i po změně vývojového týmu, bylo upuštěno od varianty vkládat do simulace ručně vyhotovené kresby a jako nejvhodnější řešení bylo vybráno vytvoření veškeré grafiky pomocí programu 3D Studio Max. Model virtuální dílny a následné renderování obrázků, které byly použity jako podklady pro grafické prostředí dílny, vytvořil student fakulty strojní Bc. Jiří Polcar. Na Obr. 5.2 a Obr. 5.3 jsou

vidět pohledy do zkušební verze virtuální dílny. Po ukončení zkušební fáze bylo rozhodnuto změnit grafický vzhled dílny za poněkud realističtější modely.



Obr. 5.3 Detail virtuálního stroje

5.3 Studijní část simulace

Hlavním účelem simulace je seznámení uživatele s přítomnými stroji. Toho bude docíleno jednak tím, že při každé příležitosti dostane uživatel několik informací o stroji, kterým se právě zabývá. Dále bude mít možnost samostatně prozkoumávat jednotlivé části strojů, jejich funkci a hlavní vlastnosti. Také bude mít neustále k dispozici virtuální skripta. Tato skripta může kdykoliv otevřít a prostudovat si více do hloubky téma, které ho bude zajímat.
Velmi kvalitním zdrojem informací, vhodných pro tuto aplikaci se jeví být skripta „Technologie obrábění a montáže“ od prof. Ing. Františka Sovy, CSc., která byla vydána Západočeskou univerzitou v Plzni v roce 2001. Tato skripta obsahují 267 stran poučného textu, který je rozdělen do několika částí podle popisované výrobní technologie. Pro naše potřeby bude postačovat několik kapitol, týkajících se problematiky strojů, nacházejících se ve virtuální dílně. Tyto vybrané stránky byly digitalizovány a vloženy do zmiňovaných virtuálních skript.
Další otázka, nad kterou je třeba se zamyslet, je jakým způsobem uživateli předložit virtuální skripta ke studiu. Jak již bylo zmíněno v kapitole 4.1.4 Manažerský simulátor MS1, Ing. Miroslav Suchý použil ve své práci informační tablet založený na podobném modelu, který se zobrazí kdykoliv při stisku klávesy F1. Autor této práce se ale domnívá, že bude vhodnější ponechat funkční klávesu F1 k účelu, který od ní očekává většina běžných uživatelů, tedy k zobrazení nápovědy. Skripta proto budou mít podobu virtuální knihy, kterou uživatel nalezne na scéně s nástěnkou.

5.4 Praktická část simulace

Praktická část simulace je laděna spíše oddechově. Zde si může student zábavnou formou nově nabyté znalosti vyzkoušet.

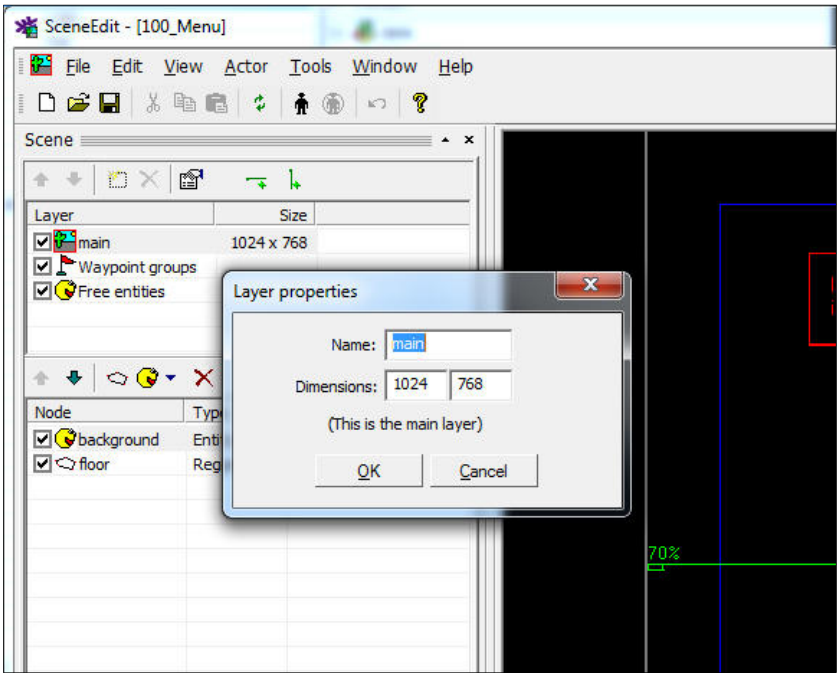
Na virtuálních strojích: nástrojářské frézce, sloupové vrtačce a univerzálním hrotovém soustruhu si student může vyzkoušet výrobu jednoduché součásti, podle dodaného výrobního postupu. Program sám povede studenta celým procesem výroby. Obrábění dílů na virtuálních strojích pomůže studentovi lépe pochopit nové znalosti, které nabyl během studijní části simulace.

6 Tvorba grafického prostředí simulace

V této části bude podrobněji popsán způsob tvorby grafického prostředí virtuální dílny a jednoho z přítomných strojů. Způsob tvorby ostatních strojů je analogický s uvedeným postupem.

6.1 Postup při vytváření scén

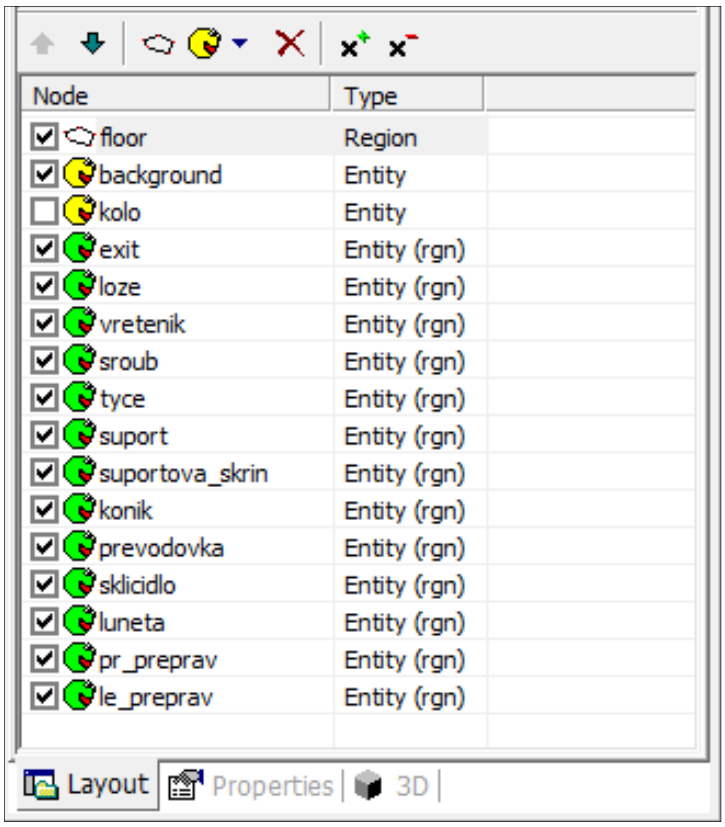
Proces vytvoření scény probíhá podle následujícího scénáře. Novou prázdnou scénu vytvoříme kliknutím pravým tlačítkem myši na složku scenes ve středním panelu okna Project manageru a následným vybráním volby Add scene scénu vytvoříme. V následně zobrazeném okně je nutné zadat název a grafické rozlišení scény.
Protože všechny šablony scén, které jsou při vytváření nové scény k dispozici, jsou pouze v rozlišení s úhlopříčkou 4:3, je nutné po vytvoření nové scény nejdříve tyto parametry pomocí tlačítka Layer proprieties (Obr. 6.1) ručně upravit na hodnotu 1280x720, což je rozlišení, které bude použito na všechny scény v simulaci.



Obr. 6.1 Scene Editor – úprava rozlišení scény

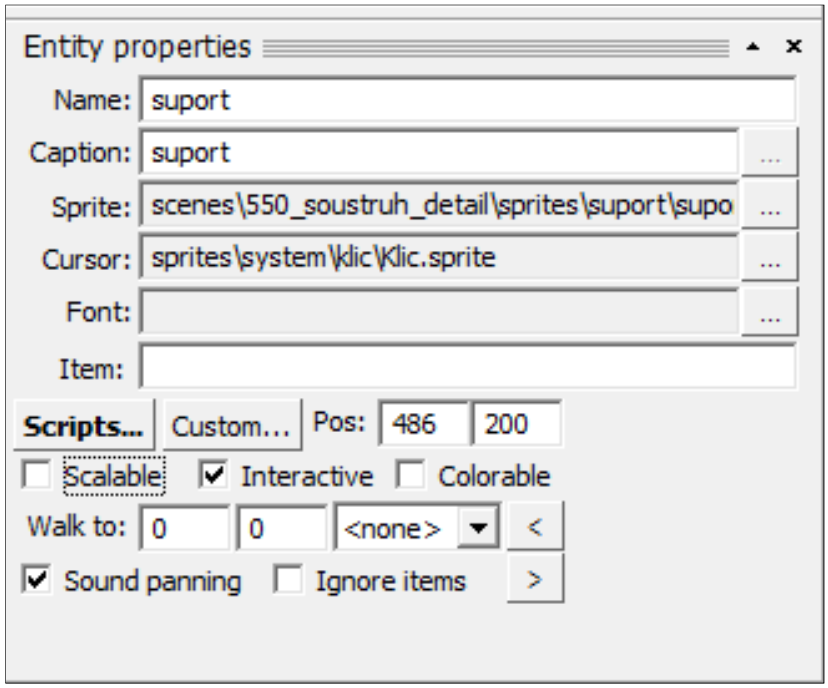
Po nastavení správného rozlišení můžeme přistoupit k importu pozadí scény a nastavení aktivních oblastí scény. Aktivní oblasti používané ve WME můžeme rozdělit do několika základních skupin. Jsou to oblasti typu region, region entity a sprite entity.
Oblast typu region je určena k vymezení prostoru, po kterém se může pohybovat hlavní postava, jinými slovy touto oblastí definujeme podlahu scény. Simulace virtuální dílna je koncipována tak, že se děj odehrává z pohledu uživatele, proto by se mohlo zdát, že tento region nebude pro tvorbu potřeba. Opak je ale pravdou. Ve WME platí jedno nepříjemné omezení, které způsobilo problém hned na začátku tvorby programu. Veškeré zobrazované texty může pronášet pouze hlavní, nebo některá z vedlejších postav. Odstraněním podlahy a hlavní postavy proto vyvstal problém, že nebylo možné na scéně zobrazovat žádné texty. Po několika marných pokusech toto omezení obejít různými úpravami řídících skriptů byl nakonec tento problém vyřešen navrácením regionu podlahy a hlavní postavy zpět do levého horního rohu scény. Této postavě byl vhodným zásahem do jednoho z hlavních řídících

skriptů znemožněn pohyb a následně byla postava skryta za grafické pozadí scény. Z této pozice může nyní postava předávat uživateli programu všechny potřebné informace a zároveň nemůže dojít k nežádoucímu ovlivnění dějové linie simulace.
Další významnou oblastí scény je oblast s názvem Region entity. Tato oblast je v seznamu entit, který se nachází v levé části Editoru scén, označena ikonou zelené barvy (Obr. 6.2). Oblasti tohoto typu mohou mít specifické vlastnosti a vlastní skripty, ale nemohou mít přiřazen žádný sprite. Entity tohoto typu byly použity především k popisu jednotlivých částí strojů.



Obr. 6.2 Scene Editor – seznam aktivních oblastí

Po vložení aktivní oblasti na scénu a uzpůsobení jejího tvaru požadavkům je potřeba následně každé z nich zadat v oblasti nazvané Entity proprities nacházející se v levé dolní části okna editoru jejich název (name). Tento parametr označuje daný objekt pro programování skriptů, proto musí být jeho hodnota v dané scéně unikátní. Další volitelné vlastnosti jsou Caption, neboli popisek, který se zobrazí vedle kurzoru myši v okamžiku, kdy uživatel nad objekt najede kurzorem myši. Do položky Cursor může být uložena alternativní podoba kurzoru, zobrazovaného při interakci s daným objektem a konečně parametrem Font lze ovlivnit písmo zobrazovaného textu. Parametr Sprite a Item nejsou v případě objektu typu Region entity aktivní. Pod těmito položkami se nachází další důležité tlačítko, pomocí kterého můžeme připojit k danému objektu již existující, nebo vytvořit nový řídící skript. Skutečnost, že je ke zvolenému objektu přiřazen skript dává WME najevo zvýrazněním popisu tlačítka Scripts tučným písmem (Obr. 6.3).

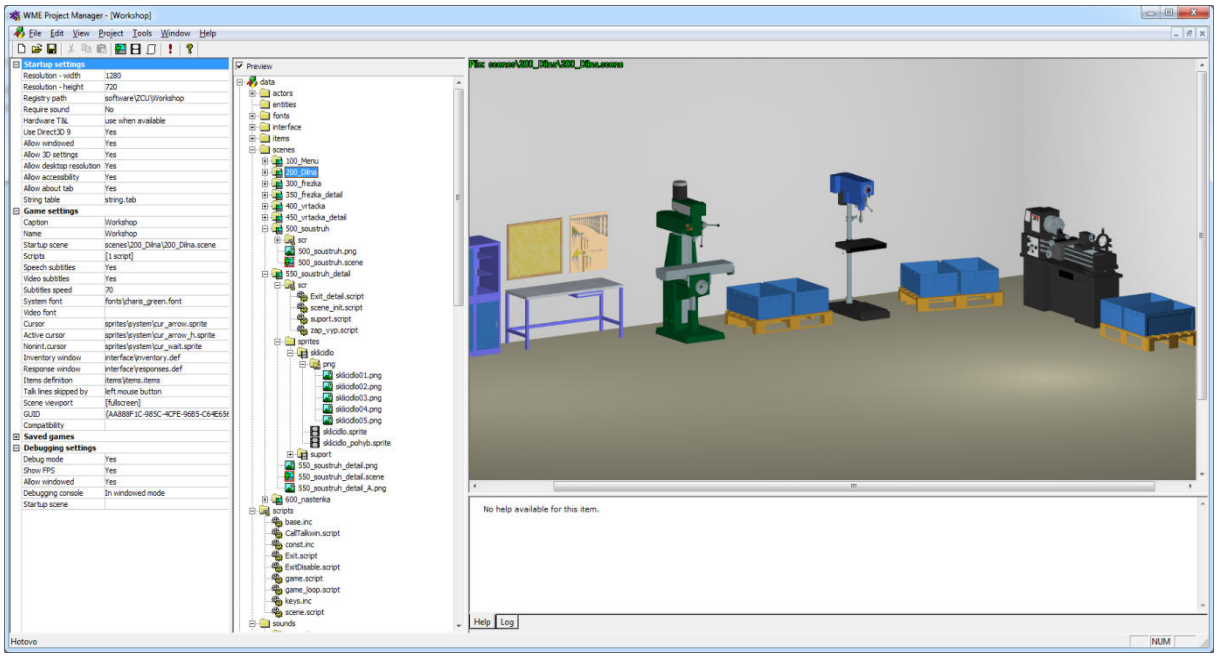


Obr. 6.3 Scene Editor – nastavení vlastností objektu

Další hojně využívaná skupina oblastí se nazývá Sprite entity. Ve zmiňovaném seznamu aktivních oblastí (Obr. 6.2) je označen ikonami žluté barvy. Postup tvorby těchto oblastí je podobný jako při vytváření předchozích entity typu region s tím rozdílem, že je možné tomuto objektu navíc přiřadit také hodnotu Sprite a Item. K parametru Sprite může být propojen statický obrázek, jak je tomu běžné v případě vkládání pozadí scén, nebo pohyblivý objekt, který se zde skládá ze sekvence několika obrázků vzájemně propojených pomocí modulu Sprite Editor. Propojení objektu s požadovaným sprite se provádí zadáním relativní cesty k souboru vkládaného objektu. Na Obr. 6.3 je patrno propojení suportu soustruhu ke spritu, zobrazujícího jeho pohyb během soustružení.
Každý z výše uvedených objektů lze v seznamu aktivních oblastí jednoduše aktivovat, nebo deaktivovat zaškrtnutím příslušného políčka nacházejícího se vedle jeho názvu, čímž se objekt dočasně odstraní ze scény. Tuto volbu lze později podle potřeby kdykoliv změnit pomocí příkazu přímo ze skriptu.

6.1.1 Celkový pohled na dílnu

První scéna, která byla vytvořena, je zároveň první scéna, se kterou se uživatel po spuštění simulace setká. Scéna ukazuje celkový pohled do dílny (Obr. 6.4) a zároveň slouží jako propojovací článek mezi jednotlivými studijními pohledy na jmenované stroje. Další významný prvek scény je stůl a nástěnka, kde může uživatel nalézt virtuální skripta s podrobnými informacemi o přítomných strojích a o způsobu práce na nich. Kolem každého ze strojů a také kolem stolu s nástěnkou je vytvořena oblast typu Region entity. Každá z těchto oblastí má přiřazený skript s příkazem ke změně scény na příslušný stroj, který se aktivuje levým kliknutím na tuto oblast.



Obr. 6.4 Celkový náhled dílny

6.2 Tvorba soustruhu

Soustruh je jedním z nejběžnějších strojů používaných ve strojírenském podniku, proto bylo zařazení právě tohoto stroje do simulace jasnou volbou. Hlavní myšlenkou, která tvorbu virtuální dílny doprovázela již od samého počátku, bylo vytvoření simulace, která bude rozdělena do dvou režimů, a to do režimu studijního a režimu virtuální praxe. Hlavní účel studijního režimu spočívá v tom, že uživatel může procházet mezi jednotlivými stroji, prozkoumávat jejich části a zjišťovat informace o jejich funkcích. V režimu virtuální praxe si poté může student vyzkoušet funkci strojů během výroby demonstračního dílu. Možnost přechodu do tohoto režimu se automaticky aktivuje v okamžiku, kdy uživatel sebere ze scény nástěnka technologický postup výroby ukázkového dílu.
V následujícím textu bude popsán postup práce při vytváření jedné ze scén simulace, kterou je scéna s univerzálním hrotovým soustruhem. Tvorba ostatních scén pak probíhala analogicky s uvedeným postupem.

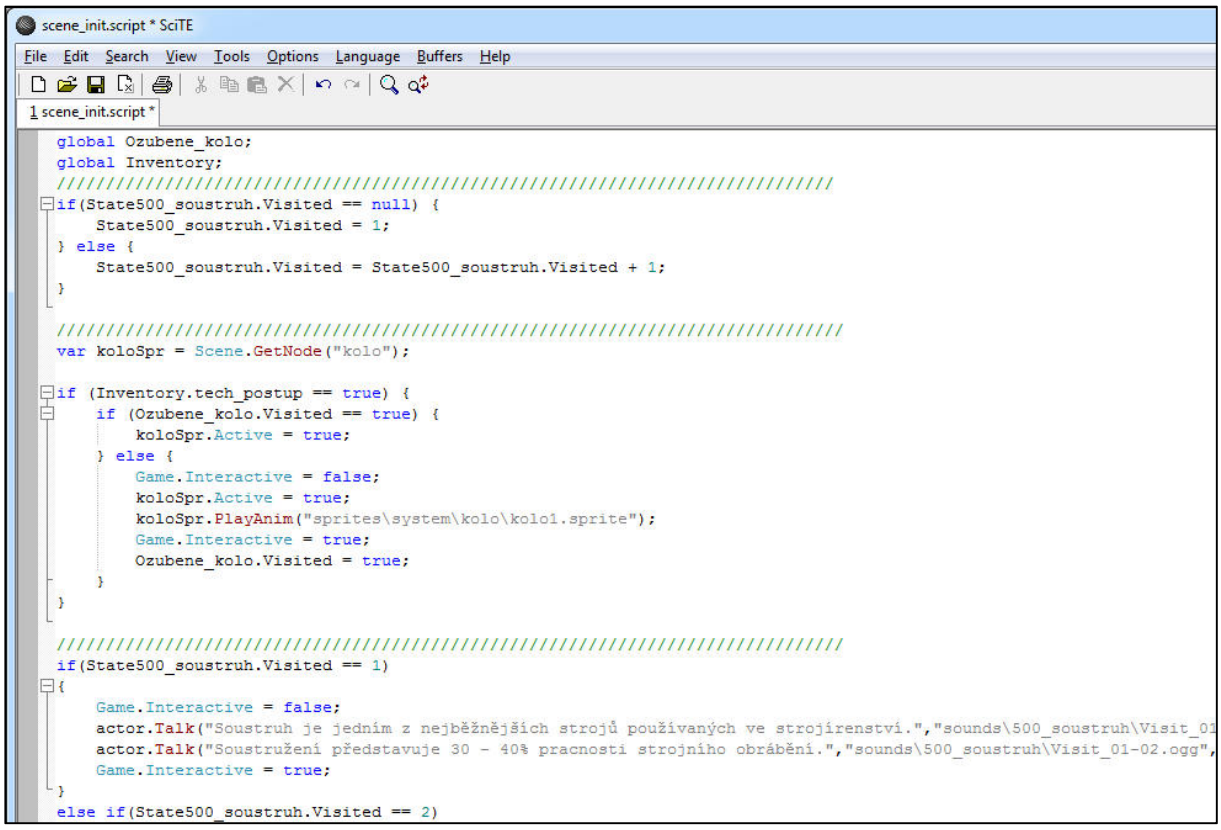
6.2.1 **Příchod na scénu**

Pro změnu z celkového pohledu na dílnu na studijní pohled na stroj provede uživatel kliknutím na obrázek příslušného stroje. Při změně scény vzniká dobrá příležitost k tomu, sdělit uživateli několik faktů o stroji, ke kterému právě přistoupil. Aby předkládané informace měly co největší efekt, nesmí být uživatel hned při první návštěvě stroje informacemi zahlcen. Z tohoto důvodu byl raději zvolen model, kdy je uživateli při každém přístupu předloženo jen několik informací o problematice k danému stroji. Bylo tedy nutné zajistit, aby se již zobrazené texty neopakovaly a uživatel dostal pokaždé nové informace. Z uvedeného vyplývá, že program musí dokázat vyhodnotit pořadí návštěvy uživatele u každého stroje a podle toho také uzpůsobit zobrazovaný text.
K uchování nejrůznějších vlastností scény slouží ve WME standardně proměnná s názvem příslušné scény. Další velice zajímavou vlastností tohoto enginu je skutečnost, že všechny proměnné použité ve skriptech jsou ve výchozím stavu typu záznam, takže uchovávání všech hodnot náležících k dané scéně zůstává vždy maximálně přehledné.
Proměnná určená ke kontrole návštěvy scény má ve výchozím stavu tvar *NazevSceny.Visited.* Tato proměnná je typu boolean, může proto nabývat pouze hodnot *true* nebo *false*. Pro tuto

simulaci je však potřeba, aby proměnná nabývala více hodnot, proto byla převedena na proměnnou typu integer a následně pomocí jednoduché sekvence příkazů počítán přesný počet návštěv. Výsledný kód vypadá následovně:

```
if(NazevSceny.Visited == null)
{
        NazevSceny.Visited = 1;
} else {
        NazevSceny.Visited = NazevSceny.Visited + 1;
}
```

Nyní je pouze potřeba otestováním hodnoty proměnné *NazevSceny.Visited* zjistit pořadí návštěvy dané scény a přiřadit text, který bude následně zobrazen.
Další věc, která musí být při příchodu na scénu otestována, je skutečnost, jestli má být uživateli zpřístupněn režim obrábění, či nikoliv. Režim obrábění bude uživateli zpřístupněn v okamžiku, kdy sebere ze scény s názvem nástěnka předmět technologický postup. Proto musí být testována přítomnost tohoto technologického postupu v inventáři.



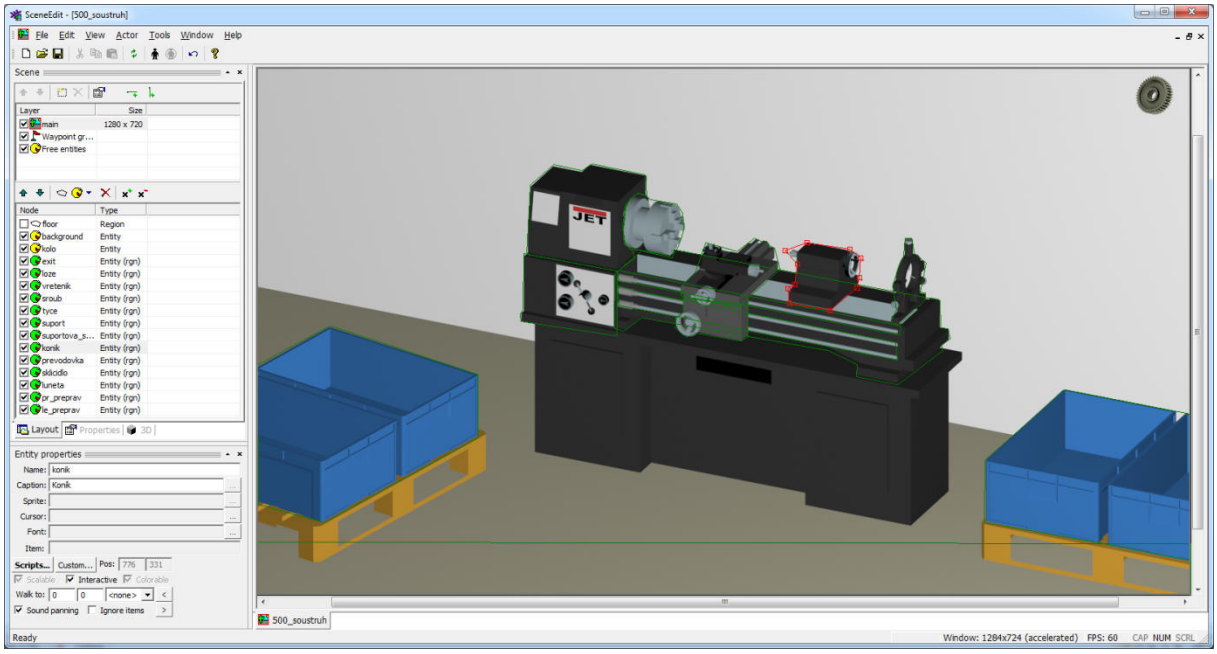
Obr. 6.5 Skript – příchod na scénu soustruh

Aby byla vyloučena možnost, že si student tlačítka pro přechod do režimu obrábění nevšimne, obsahuje skript také příkaz na přehrání animace kolo1.sprite, která ho na přítomnost tohoto tlačítka náležitě upozorní. Tato animace se přehraje během celé simulace pouze jednou. Výsledný kód skriptu je patrný z Obr. 6.5.

6.2.2 Studijní režim soustruhu

Pohled na stroj ve studijním režimu je situován z dostatečné vzdálenosti, aby byly dobře vidět všechny části stroje. Při vytváření tohoto pohledu byl do scény umístěn obrázek se strojem,

sloužící jako pozadí (Obr. 6.6). Jak již bylo zmíněno v kapitole 6.1 Postup při vytváření scén, je nezbytné, aby entita s pozadím byla umístěna až za region podlaha, protože objekt, který je v seznamu umístěn níže, je zobrazován až po svém předchůdci a proto jej na scéně překryje.
Následně byly postupně vkládány ve vhodném pořadí na jednotlivé části stroje objekty typu region entity. Jejich tvar byl upraven tak, aby odpovídal tvarům části stroje a byly zadány hodnoty položek Name a popisek Caption.



Obr. 6.6 Soustruh – studijní režim

Po označení všech částí stroje byly ke všem následně přidány skripty, které po kliknutí na objekt zajistí zobrazení správné informace o dané části stroje. Všechny tyto skripty jsou uloženy ve složce scr, která je umístěna ve složce příslušné scény. V případě popisovaného soustruhu bude tedy mít výsledná adresa tvar „\scenes\500_soustruh\scr“.
Každý skript definuje akce, které proběhnou po kliknutí levým, nebo pravým tlačítkem myši. V našem případě se po kliknutí levým tlačítkem zobrazí pouze základní informace o dané části stroje a po kliknutí pravým tlačítkem budou zobrazeny podrobnější informace.

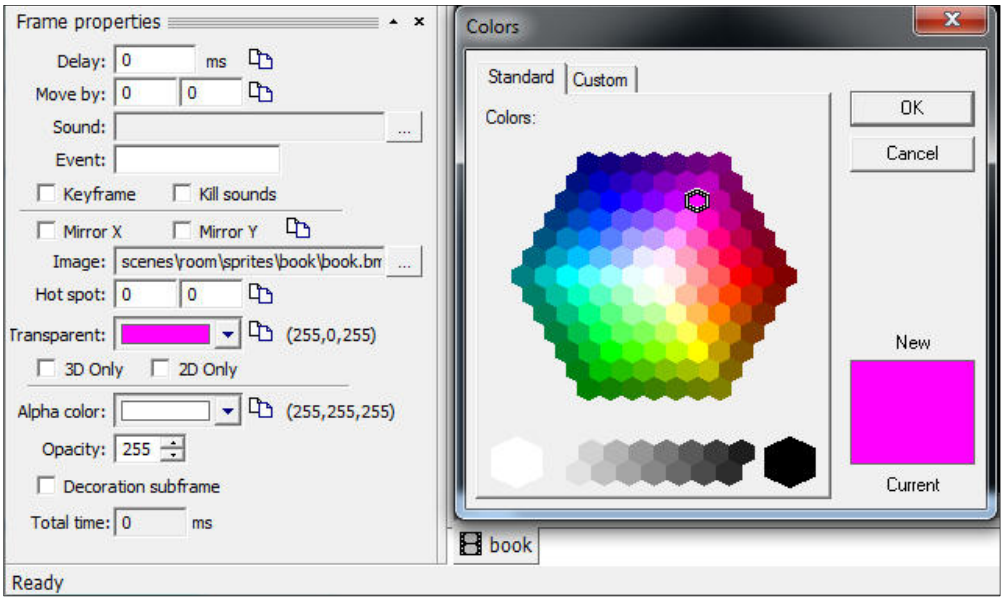
6.2.3 **Praktický režim soustruhu**

Režim obrábění je přístupný přes tlačítko ozubené kolo, které je umístěné v pravém horním rohu scény studijního pohledu. Toto tlačítko se zobrazí až v momentě, kdy uživatel sebere ze scény nástěnka předmět výrobní postup.
Po kliknutí na ikonu ozubeného kola se změní scéna. Pohled na soustruh je nyní z menší vzdálenosti, aby byly dobře přístupné ovládací prvky stroje. Také úhel pohledu se změní tak, aby více vyhovoval práci na stroji. U jednotlivých částí stroje již není jejich popis, jak tomu bylo u předcházejícího studijního režimu. Po kliknutí na příslušnou entitu bude pak tato součást použita k účelu, ke kterému je určena. U soustruhu je možné roztočit vřeteno oběma směry, ale pouze jeden je ten správný. Dále je možné pohybovat suportem s nožovou hlavou, čímž je evokována představa obrábění dílce.
Přestože je v technologickém postupu uvedeno, že díl má být během soustružení podepřen koníkem, je uživateli umožněno díl obrábět také s upnutím letmo, tedy bez využití koníka. Při tomto postupu však prudce stoupá pravděpodobnost výroby zmetku až na hodnotu 70%.

Uživatel je po té vyzván, aby znovu prostudoval výrobní postup, a nepovedený kus musí vyrobit znovu.
Velice výchovně působí i skutečnost, že nelze odejít od zapnutého stroje. Při pokusu opustit scénu bez toho, aby byl před tím vypnut stroj, bude uživatel upozorněn na chybu, kterou se pokusil udělat a odchod mu nebude umožněn, dokud ji nenapraví a stroj nevypne.

6.2.4 Tvorba pohyblivých součástí

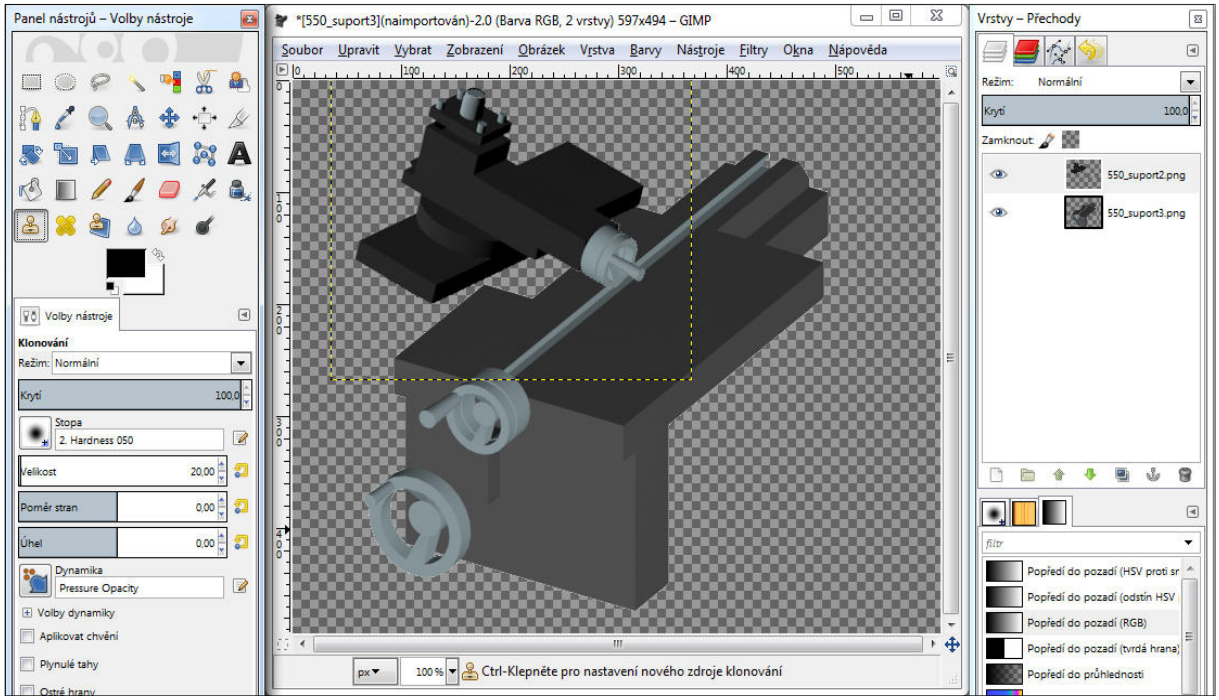
Veškeré pohyblivé části a animace jsou do scén přidávány jako objekty typu Sprite entity. Pohyb těchto částí po pozadí scény musí být definovány s vysokou přesností, aby pohyb souhlasil s polohou vodících prvků na podkladové scéně a pohyb vypadal co nejrealističtěji. I ty nejdrobnější nepřesnosti se během pohybu ihned projeví jako chvění pohybujícího se objektu, nebo trhavý a neplynulý pohyb.
Výsledný pohyblivý sprite se skládá ze sekvence po sobě jdoucích jednotlivých obrázků. Tyto jednotlivé obrázky je nutné nejdříve připravit tak, aby bylo možné je následně do grafiky vložit a obrázky překrývaly scénu pouze podle svého tvaru. Ve WME jsou dva způsoby, jak toho lze dosáhnout. První způsob využívá specifické vlastnosti WME. Ve Wintermute Engine lze definovat barvu, kterou bude engine ignorovat a nebude jí vůbec zobrazovat. Ve výchozím nastavení je touto ignorovanou barvou purpurová (Magenta) která je označena v barevném modelu RGB kódem 255, 0, 255. Tuto barvu lze však při tvorbě sprite libovolně zaměnit za jakoukoliv jinou (Obr. 6.7).



Obr. 6.7 Sprite editor – Volba neviditelné barvy

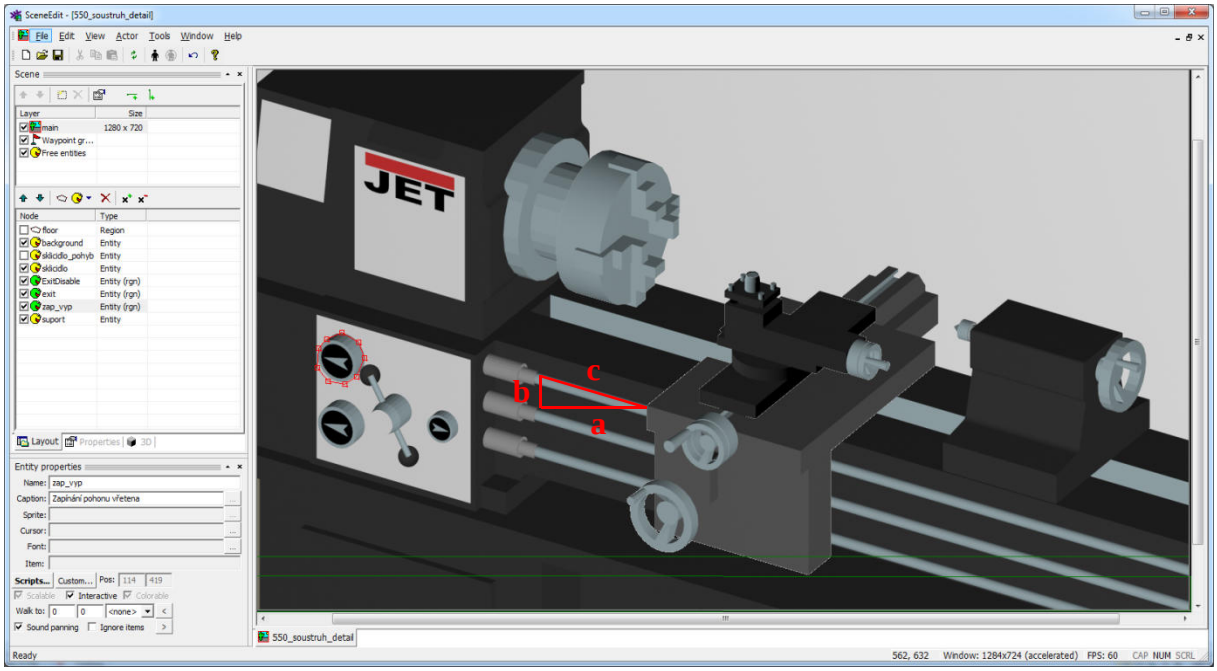
Druhou možností je využít tzv. alfa kanál, obsažený v obrázcích formátu PNG. Tento kanál určuje hodnotu průhlednosti pixelu. To znamená, že pokud obrázek s definovaným alfa kanálem překrývá jiný obrázek, bude obrázek na pozadí zobrazen s intenzitou danou hodnotou průhlednosti obrázku v popředí.
Základní grafika vyrenderovaná z modelu dílny z programu 3D studio Max, kterou vytvořil Bc. Jiří Polcar obsahovala různé nežádoucí pozadí, jako jsou například stěny dílny, a proto nebylo možné ji přímo použít v programu. Ze všech obrázků pohyblivých částí bylo nejdříve nutné odstranit veškeré pozadí a nahradit ho průhledným alfa kanálem, aby nedocházelo k překrytí pozadí scény v okolí pohybujícího se předmětu.
Veškeré úpravy obrázků pro tuto simulaci, včetně přidávání alfa kanálu, byly realizovány pomocí programu Gimp (Obr. 6.8). Gimp je svobodná multiplatformní aplikace, určená pro

úpravu a vytváření rastrové grafiky. Kromě široké škály rastrových nástrojů obsahuje i některé vektorové funkce, které v tomto případě ale nebudou potřeba. Gimp je dostupný zdarma včetně zdrojových kódů pod licencí GPL.



Obr. 6.8 Gimp – úprava obrázku pro sprite

Po úpravě obrázků určených pro jednotlivé pohyblivé sprite bylo možno zahájit samotnou tvorbu animací pohybu. Jako první animace byl vytvořen pohyb suportu soustruhu. Protože pohyb suportu byl vytvořen taktéž pomocí Sprite editoru a je tudíž tvořen obrázky pohybujícími se v určitém směru pomocí přesně definovaných souřadnic každého snímku, bylo nezbytné tyto souřadnice stanovit s dostatečnou přesností, aby se pohyb suportu shodoval s umístěním pojezdů na loži stroje.



Obr. 6.9 Scene editor – pohyb suportu

Vzdálenost, kterou musí při pohybu suport urazit je zvýrazněna na Obr. 6.9. Jak je dále z obrázku patrné, pohyb suportu po dráze c lze matematicky rozložit do směru *x* definovaného vzdáleností *a* a směru *y*, kde je definován úsečkou *b*. Vzájemné závislosti těchto vzdáleností lze definovat vztahem.

$$c = \sqrt{a^2 + b^2}$$

Počet snímků, které budou potřeba na přehrání animace, pak můžeme vyjádřit pomocí vztahu.

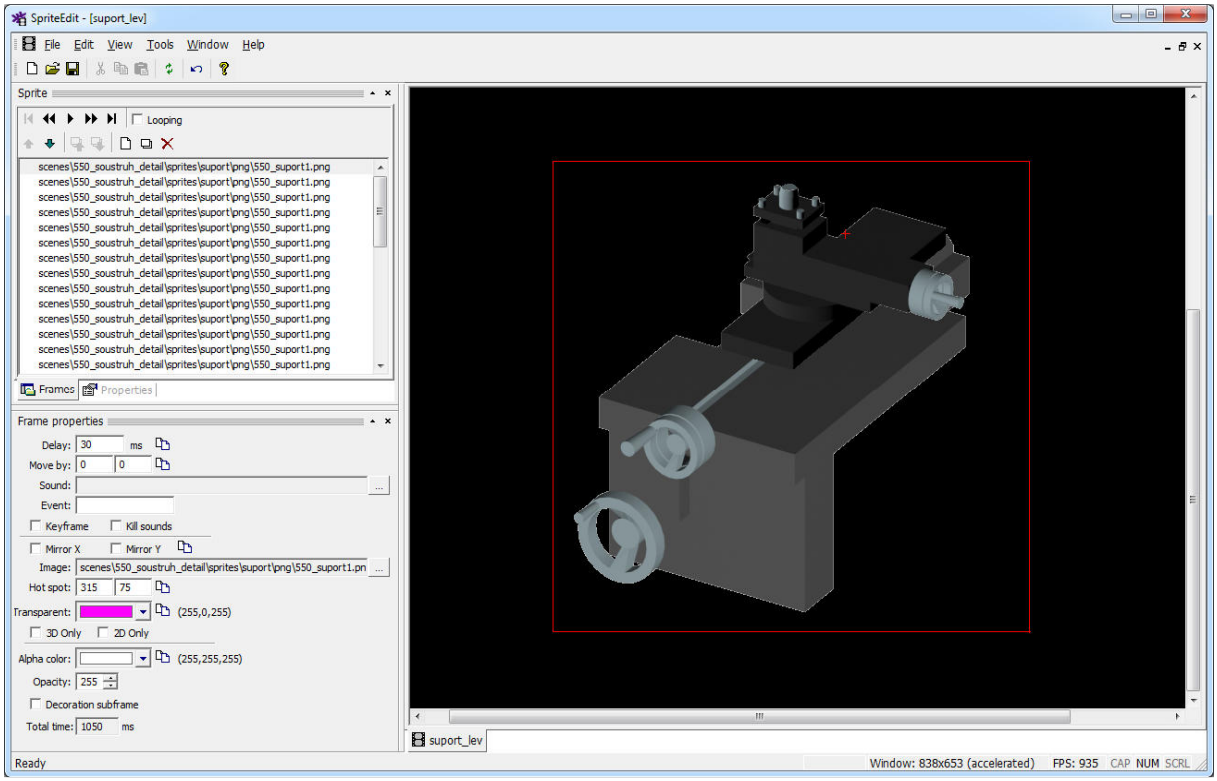
$$n = \frac{\sqrt{a^2 + b^2}}{k}$$

Kde hodnota *n* je počet snímků, ze kterých se bude celý pohyb skládat a proměnná *k* představuje požadovanou vzdálenost, kterou má suport urazit během jednoho snímku.
Čas, na který se mají jednotlivé snímky zobrazit, pak zjistíme podle následujícího vztahu.

$$f = \frac{k \cdot t}{\sqrt{a^2 + b^2}}$$

Kde proměnná *f* představuje čas zobrazení jednotlivého frame, neboli snímku animace a hodnota *t* představuje celkovou délku animace.
Podle těchto vztahů byla následně pomocí MS Excel vytvořena tabulka. Tato tabulka po zadání základních údajů o souřadnicích počáteční polohy objektu, souřadnicích koncové polohy objektu, požadovaného počtu snímků a celkové doby trvání animace sama vypočítá všechny potřebné hodnoty pro všechny snímky animace, které je při tvorbě sprite nutné zadat. Pro zjištění počátečních a koncových souřadnic pohybu bylo využito schopnosti Sprite editoru zobrazovat přesné souřadnice objektů umístěných do scény. Suport byl vložen do výchozí polohy a poté myší přetažen do požadované koncové polohy. Takto získané souřadnice byly následně použity s využitím dříve uvedených matematických vztahů k přesnému určení polohy jednotlivých snímků animace.

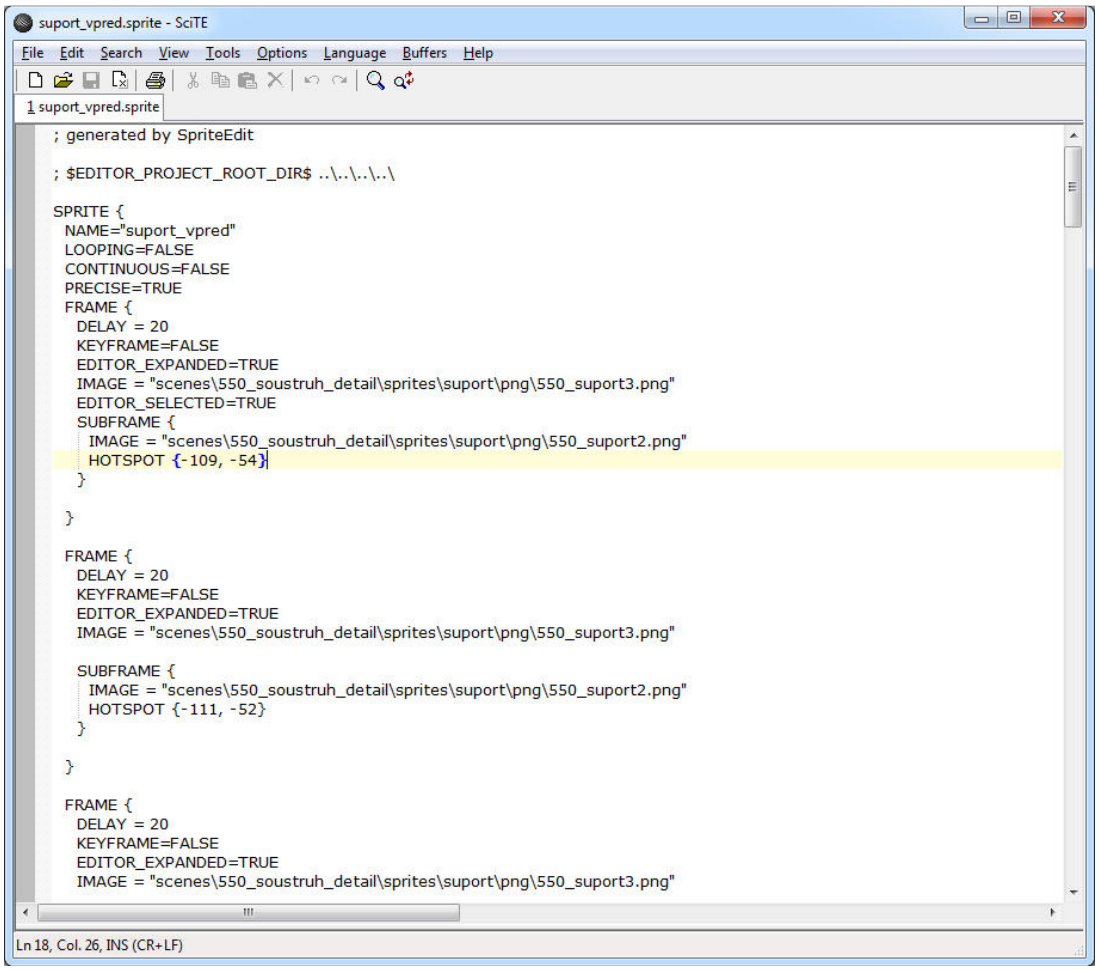
Po zjištění souřadnic všech snímků následovalo samotné vytváření sprite. Jednotlivé snímky byly ve správném pořadí vkládány pomocí Sprite Editoru do animace a ke každému z nich byly následně zadány souřadnice a čas zobrazení v milisekundách. Pořadí zobrazování snímků lze měnit jeho přesunutím v seznamu pomocí šipek v pravé horní části editoru. Při tvorbě složitějších animací je také možnost vkládat ke každému snímku tzv. subframe, neboli podřízené snímky. Každý z těchto snímků má vlastní souřadnice vztažené k počátečnímu bodu a jsou na scéně vykreslovány pouze po dobu zobrazení jejich rodičovského snímku. Složitost vytvoření takovéto animace potom stoupá exponenciálně s počtem použitých subframe snímků. Systém s využitím subframe snímkování byl použit na scéně soustruhu pouze při posuvu příčného suportu vpřed, kdy rodičovský snímek byl statický podélný suport a subframe pohyblivý příčný suport. Sprite znázorňující pohyb obou suportů v souladu s podélnou osou stroje, který je vyobrazen na Obr. 6.10, je již výsledkem spojení příčného a podélného suportu v programu Gimp do dvou vrstev jednoho obrázku. Proto zde nebylo zapotřebí použít složité výpočty sdružených pohybů, jak by tomu bylo v případě využití subframe snímkování.



Obr. 6.10 Sprite editor – suport

Hlavní výhoda tvorby animace pomocí Sprite editoru je skutečnost, že většina běžných operací lze provádět pohodlně pouze pomocí myši a výsledek je okamžitě zobrazován v náhledovém okně editoru. Při vytváření delšího přímočarého pohybu se však stává práce jednotvárným a zdlouhavým opakováním stejné sekvence pohybů myši a zadávání různých číselných hodnot. V této opakované činnosti se programátor zakrátko přestane orientovat, a proto zde enormně narůstá riziko vzniku chyby.

Z tohoto důvodu autor při vytváření těchto animací přistoupil k poněkud netradiční metodě, a sice k přímé editaci souboru sprite pomocí běžného textového editoru (Obr. 6.11). Tento způsob programování je sice v dané situaci rychlejší a přehlednější, ale také zde hrozí určitá nebezpečí. Hlavní nebezpečí tohoto způsobu editace spočívá ve skutečnosti, že specifická syntaxe souboru sprite není podporována žádným z běžně dostupných editorů, a proto nedochází ke zvýraznění příkazových vazeb, ani zde není funkční detekce překlepů a chyb. Může se tak velice snadno stát, že opomenutí jedné závorky, nebo jiná drobná chyba bude mít za následek nefunkčnost celého souboru. Potom Sprite editor nedokáže tento soubor vůbec otevřít a zpracovat ho. Nezbývá pak jiná možnost, než poškozený soubor znovu otevřít pomocí textového editoru a chybu se pokusit najít a odstranit ručně. Nicméně při udržení maximální pozornosti a opatrnosti je tento způsob editace v daném případě mnohem přehlednější a efektivnější, než je vytváření stejné animace pomocí sprite editoru.

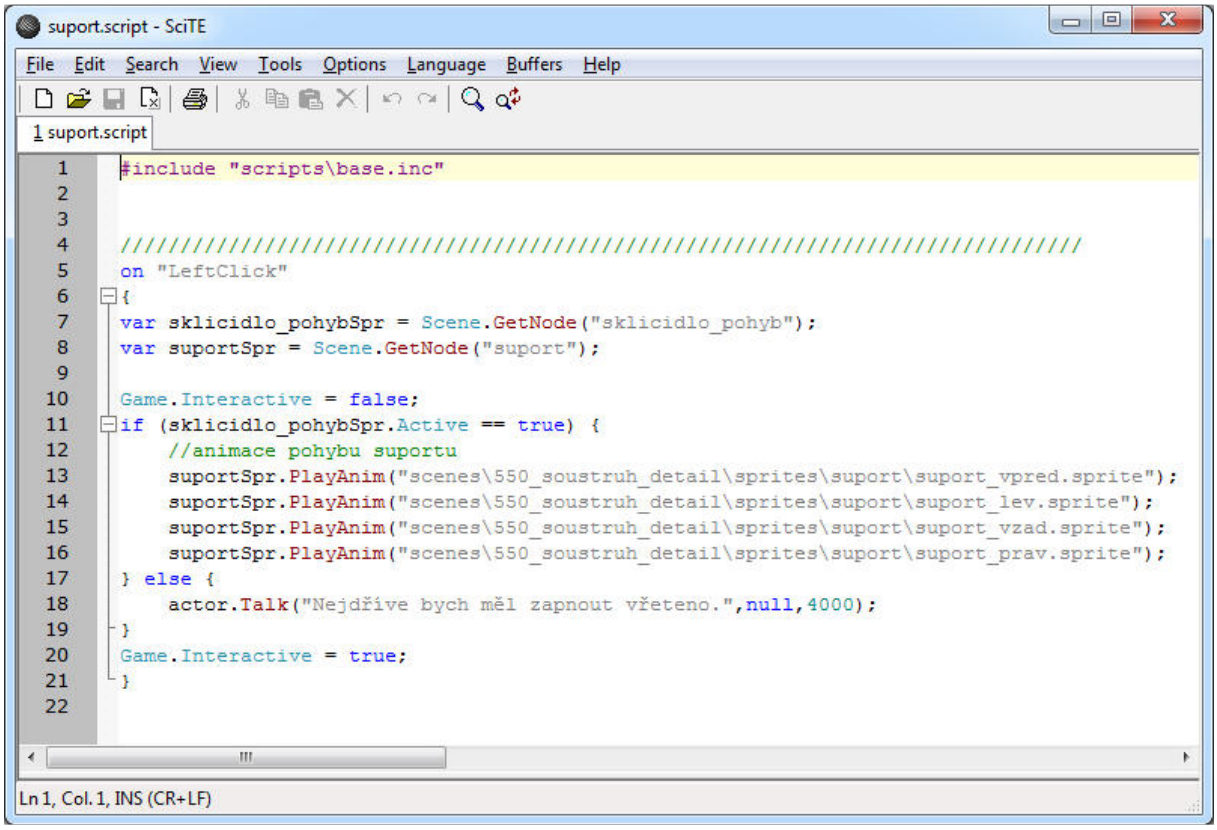


Obr. 6.11 Editace sprite pomocí textového editoru

Podobným způsobem, jaký byl popsán v předcházejících řádcích, byly vytvořeny celkem čtyři animace potřebné pro simulovaný pohyb suportu soustruhu. Animace pohybu příčného suportu vpřed, animace obou suportů zprava doleva s příčným suportem v přední poloze, posuv příčného suportu z přední polohy vzad a posuv obou suportů zleva doprava tentokrát s příčným suportem v zadní poloze. Celkem se tedy pohyb suportu skládá ze 122 frame a 52 subframe.

Na scénu se soustruhem byl následně umístěn sprite s nepohyblivým obrázkem suportu. Následný pohyb suportu je v případě potřeby vyvolán příkazem v příslušném skriptu, který

nechá postupně přehrát jednotlivé animace. Sekvence příkazů, které přehrají celou animaci s pohybem suportu, je patrná z Obr. 6.12.

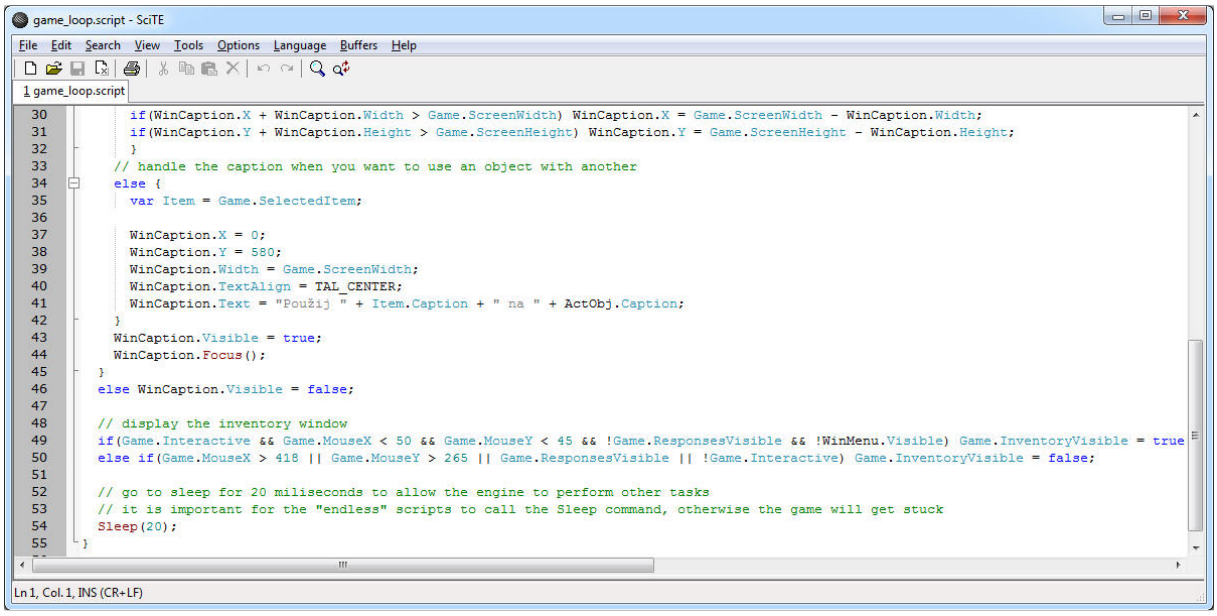


Obr. 6.12 Skript – přehrání animace pohybu suportu

Pohyb suportu je spuštěn po kliknutí levého tlačítka myši na suport (Obr. 6.12 řádek 5). Následně jsou pomocí příkazů na řádcích 7 a 8 vytvořeny proměnné *sklicidlo_pohybSpr* a *suportSpr* do kterých jsou ze scény nahrány příslušné sprite. Tato operace je nutná proto, aby bylo možné tyto sprite ovládat přímo ze skriptu. Na řádku 11 následně proběhne kontrola, jestli je vřeteno v chodu. Jestliže tomu tak je, skript postupně přehraje pomocí příkazu *PlayAnim* na řádcích 13 – 16 všechny animace. V opačném případě je uživatel upozorněn, že by bylo před zahájením soustružení nejdříve vhodné roztočit vřeteno.
Příkazy na řádcích 10 a 20 postupně pozastaví a znovu spustí interaktivitu simulace. Tyto příkazy se do skriptů při vykonávání nějaké složitější operace běžně vkládají, aby nemohlo dojít po nechtěném kliknutí k zablokování programu.

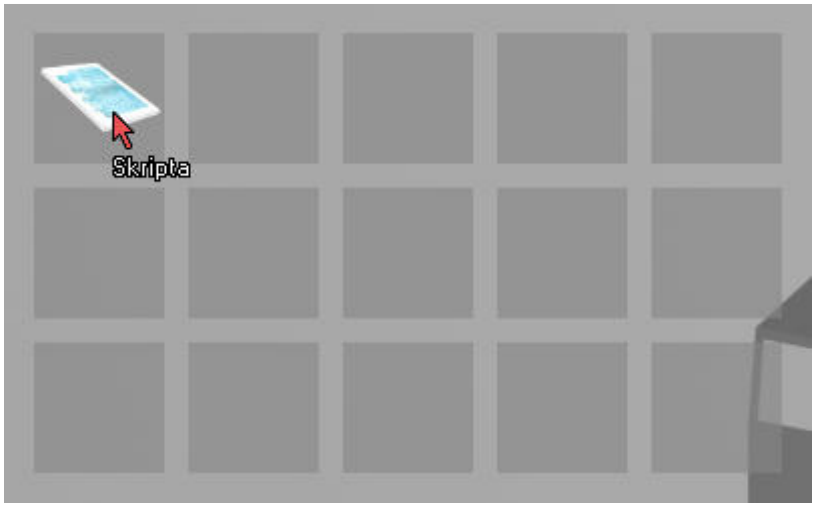
7 Tvorba inventáře

Během simulace je potřeba mít k dispozici určitý prostor, kde budou uloženy předměty, které budou uživateli neustále k dispozici. K tomuto účelu slouží inventář. V inventáři jsou uloženy všechny předměty sebrané ze scény. Tyto předměty zde mohou být prozkoumávány, kombinovány mezi sebou, nebo odtud mohou být použity na jiné objekty nacházející se na scéně. Tento inventář má formu okna, které je umístěno do levého horního rohu obrazovky.
Původní inventář, který byl k dispozici jako šablona přímo ve Wintermute Engine, nebyl příliš vyhovující, protože byl umístěn po celém horním okraji a reagoval na kurzor myši při najetí na horní hranu obrazovky. Toto umístění inventáře nebylo příliš vhodné, protože ikona ozubeného kola, která umožňuje přístup do režimu obrábění, byla umístěna také na horní okraj obrazovky. Po najetí kurzorem na ikonu ozubeného kola se zobrazoval také inventář. Proto byl inventář přesunut do levého horního rohu. Přesunutí reaktivní oblasti bylo docíleno úpravou řídícího skriptu *game_loop.script*, kde bylo přidáno vedle testování polohy kurzoru v ose y také testování v ose x. Tyto popisované změny jsou patrné na Obr. 7.1 na řádcích 49 a 50.



Obr. 7.1 Skript – reaktivní oblast inventáře

Po úpravě polohy inventáře a oblasti, kde bude software reagovat na kurzor zobrazením inventáře, je třeba uzpůsobit také vizuální stránku inventáře. Soubor *inventory.sef* s uloženým oknem inventáře se nachází v adresáři interface. Zde byl pomocí modulu Window Editor, který je součástí WME, původní obrázek se vzhledem inventáře nahrazen jinou grafikou, která byla vytvořena opět pomocí grafického editoru Gimp. Nové okno inventáře o rozměrech 403x249 pixel má shodnou barvu i hodnotu průhlednosti jako dialogové okno. Oproti dialogovému oknu jsou zde zvýrazněna jednotlivá políčka pro umístění předmětů. Předměty jsou ukládány do matice o rozměru 5x3 políček. Po levé straně okna jsou připraveny tlačítka pro přesouvání políček v inventáři doleva a doprava pro případ, že dojde k jeho zaplnění předměty. Protože zde není předpoklad, že by během této simulace došlo k zaplnění celého inventáře, jsou nyní tato tlačítka prozatím skryta. Nicméně kdyby v průběhu případného dalšího vývoje simulace vyvstala potřeba umisťovat do inventáře další předměty, stačí tato tlačítka prostřednictvím Window editoru opět přepnout na hodnotu visible a budou opět plně funkční. Konečný vzhled inventáře je vidět na Obr. 7.2.



Obr. 7.2 Okno inventáře

Nyní je třeba zajistit správné fungování inventáře vytvořením funkčních slotů pro umístění předmětů na zvýrazněná políčka podkladové grafiky okna inventáře. K tomuto účelu slouží objekt Inventory box, který je standardně k dispozici pouze v souboru s oknem inventáře.

2 - Behavior	
Scripts	
HideSelected	**True**
Exclusive	**False**
ScrollBy	**1**
3 - Layout	
Area	**15; 15; 373; 219**
X	**15**
Y	**15**
Width	**373**
Height	**219**
Spacing	**12**
ItemWidth	**65**
ItemHeight	**65**
4 - Appearance	
Visible	**False**

Obr. 7.3 Konfigurace Inventory boxu

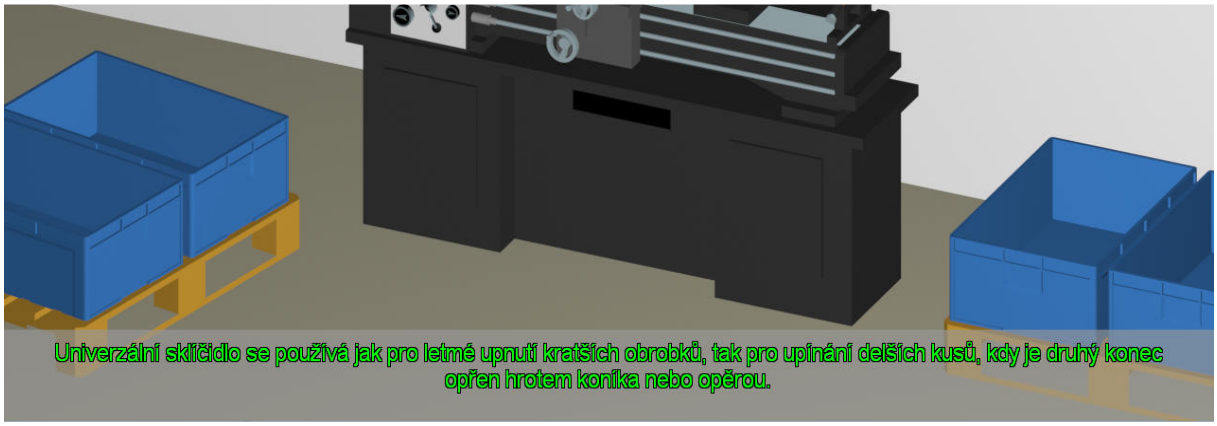
V simulaci virtuální dílny budou mít všechny náhledy předmětů určené pro zobrazení v inventáři rozměr 65x65 bodů, proto musí mít tento rozměr také sloty pro umístění předmětů v inventáři. Inventory box lze taktéž editovat pomocí Window editoru (Obr. 7.3). Jako první je nutné nastavit souřadnice pro umístění oblasti na obrazovce. Počátek okna byl posunut na hodnotu x = 15 a y = 15 bodů proto, aby vznikl na okraji okna prázdný rámeček, do kterého nebudou sloty zasahovat. Celková velikost oblasti Inventory box je 373x219. Zde jste si jistě povšimli, že se rozměr Inventory boxu neshoduje s rozměrem okna inventáře. Je to zapříčiněno právě zmiňovaným zámečkem 15 bodů kolem Inventory boxu. Rozměr jednotlivých slotů bude 65x65 bodů a rozestupy mezi sloty 12 bodů. Po zadání všech těchto náležitostí engine sám vypočítá, kolik slotů se do oblasti vejde a podle toho je zde také rozmístí.

8 Změna písma a tvorba dialogového okna

Jak již bylo uvedeno v kapitole 6.1 Postup při vytváření scén, veškeré texty, které se zobrazují v průběhu simulace, musí pronášet nějaká postava umístěná do scény. Tyto texty se ve výchozím nastavení zobrazují nad hlavami postav, což je poměrně nepřehledné. Velikost písma je navíc docela malá, což celou situaci ještě zhoršuje. Vzhledem k tomu, že v této simulaci je postava, která zajišťuje zobrazování textů, skryta za scénou v levém horním rohu, zobrazují se všechny texty nesymetricky na horním okraji obrazovky. Takto zobrazené texty jsou velice nepřehledné, a proto je takovýto způsob zobrazení zcela nepřípustný. Naštěstí existuje poměrně snadné řešení. Na stránkách výrobce Wintermute Engine je možné nalézt návod, jak vytvořit takzvanou metodu Talk, která tento problém řeší.

8.1 Tvorba metody Talk

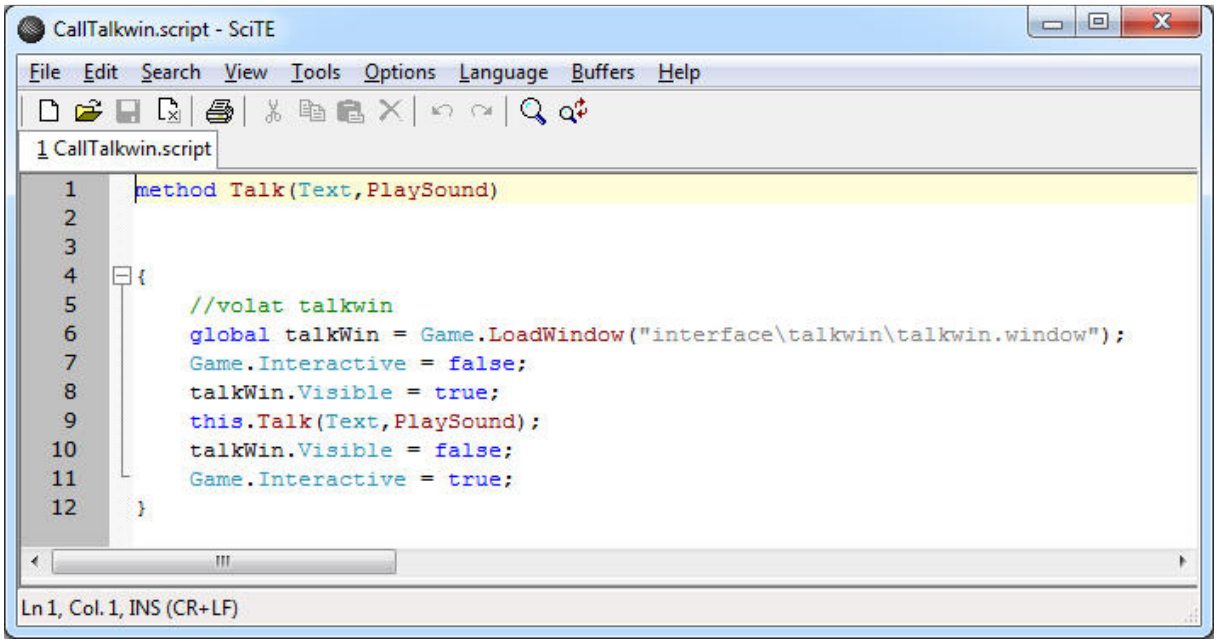
Princip metody Talk spočívá ve vytvoření podkladového okna, které je možné umístit na libovolné místo obrazovky. V tomto okně se poté bude zobrazovat veškerý text, který se na scéně objevuje. Okno se při zahájení rozhovoru automaticky otevře a po zmizení poslední zobrazované věty textu se okno opět samovolně uzavře. Volání okna bude probíhat přímo metodou Talk, kterou je třeba nejdříve modifikovat. Na Obr. 8.1 je patrný náhled hotového okna TalkWin včetně textu, jak bude zobrazeno v průběhu programu simulace.



Obr. 8.1 Zobrazení okna TalkWin během simulace

Základ textového okna je vytvořen pomocí modulu Window Editor. Nejdříve bylo ve složce interface založeno nové okno pojmenovaná TalkWin. Z tohoto nově vytvořeného okna byla následně odstraněna veškerá grafika, která byla nahrazena obrázkem ve formátu PNG, který obsahuje pouze poloprůhledný obdélník o rozměrech 1280x90, což je požadovaný rozměr nového okna TaklWin. Tento obrázek bude sloužit jako podklad pro zobrazovaný text. Následně byl pomocí myši upraven rozměr celého okna podle vloženého obrázku. Takto vytvořený panel byl umístěn na pozici, kde se má v budoucnu všechen text zobrazovat. Protože je v této simulaci počítáno s tím, že bude veškerý text ve spodní části obrazovky, byly zadány souřadnice polohy x = 0 a y = 630. Další úpravy tohoto okna pro zobrazení textu již nejsou potřeba.
Dalším krokem po dokončení okna pro text je vytvoření skriptu, který zajistí, aby se toto okno na začátku každého rozhovoru samo otevřelo a po jeho ukončení opět samo zavřelo. K tomuto účelu byl vytvořen skript s názvem CallTalkWin.script, který je umístěn v hlavním adresáři programu ve složce scripts. Obsah tohoto skriptu je patrný na Obr. 8.2. Příkaz na prvním řádku skriptu definuje metodu, kterou bude tento skript modifikovat. Po každém použití příkazu Talk v kterémkoliv skriptu, se nyní provedou příkazy na následujících řádcích.

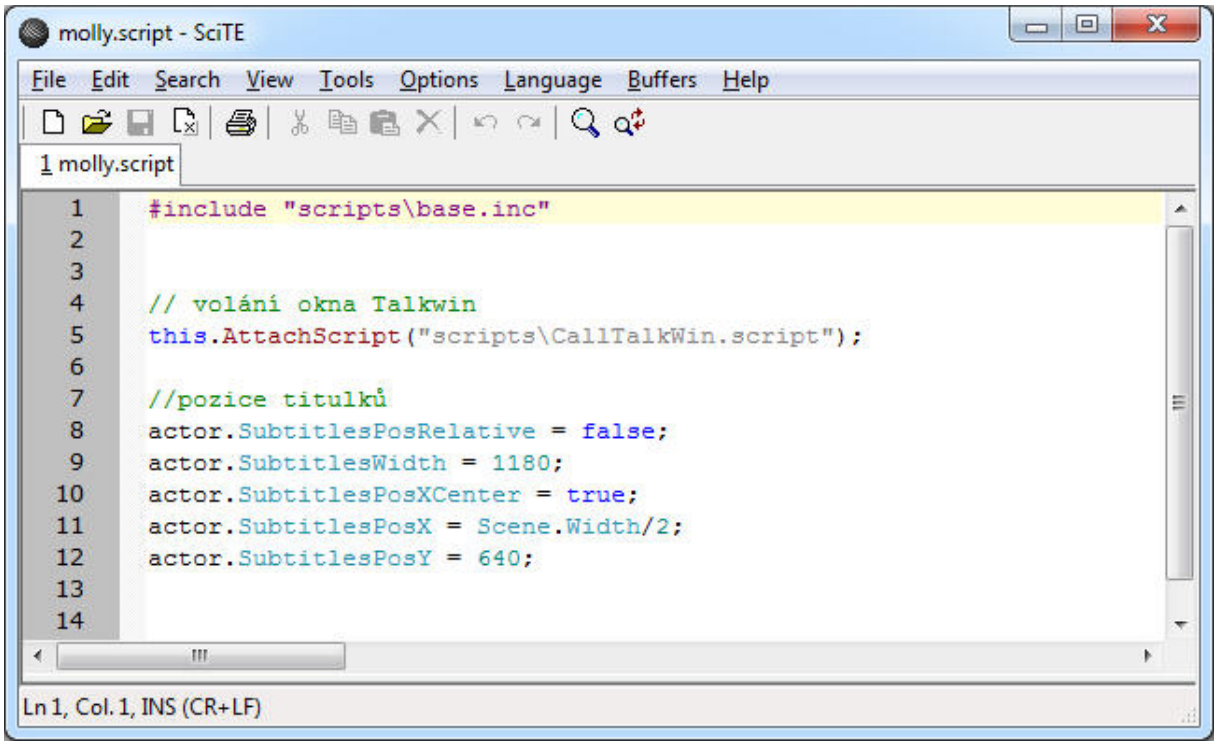
Za příkazem, který budeme modifikovat, jsou v závorce uvedeny očekávané parametry tohoto příkazu. Parametr Text stanoví, že první příkazem zpracovávaná položka bude text, který se vypíše na monitor. Protože kromě zobrazení textu je také v simulaci využito možnosti přehrání zvukového souboru, ve kterém je zobrazovaný text namluven, musí být vedle parametru *Text* zadán také parametr *PlaySound*. Oba parametry jsou vzájemně odděleny čárkou. Po deklaraci metody následuje samotné tělo skriptu.



Obr. 8.2 Metoda Talk – volání textového okna

Na řádku 6 skript nahraje do globální proměnné s názvem talkWin ze složky interface grafiku okna a připraví ji pro zobrazení. Následující příkazové řádek 7 pozastaví interaktivitu simulace, aby nedošlo k pádu aplikace po nechtěném kliknutí uživatele, a poté řádek 8 zobrazí okno talkWin na monitoru. Na řádku 9 již následuje příkaz pro vypsání textu do připraveného okna a zároveň pokud je definován zvukový soubor skriptem volajícím tuto metodu, spustí přehrávání tohoto zvukového souboru. Po zobrazení požadovaného textu a ukončení přehrávání nahrávky, skript okno talkWin opětovně skryje a uvolní zablokovanou interaktivitu programu.

Třetím a závěrečným krokem tvorby metody Talk je její připojení k postavám. V této simulaci všechny texty zajišťuje předdefinovaná postava jménem Molly, proto stačí upravit pouze jeden skript na adrese „actors\molly\molly.script“ (Obr. 8.3). V prvé řadě je nutné příkazem *AttachScript* připojit k postavě skript s v předchozím textu popisovanou metodou Talk. Následující příkazy tohoto skriptu zajišťují, aby poloha titulků byla právě v tomto okně. Příkaz na řádku 8 vypíná funkci relativní pozice titulků, která je vztažena k pozici hlavní postavy na scéně. Nyní bude pozice textu dána absolutními hodnotami uvedenými v následujících řádcích tohoto skriptu. Řádek 9 určuje maximální šířku zobrazovaného textu. Tato hodnota musí být menší, než je šířka celého okna, aby zůstával volný prostor mezi okrajem obrazovky a koncem textu. Příkazem na řádce 10 je zajištěno zarovnávání textu na střed. Řádek 11 umisťuje text titulků od středu obrazovky a řádek 12 určuje polohu titulků v ose y. Aby byl text na horní části okna správně umístěn, musí být hodnota určující polohu titulků v ose y o něco vyšší než je hodnota polohy okna TalkWin, zadaná ve Window Editoru.

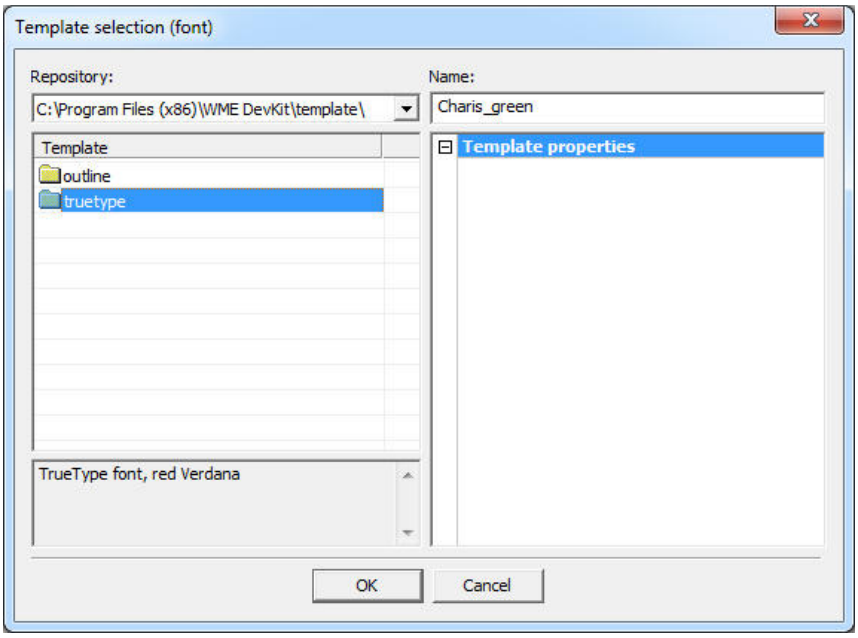


Obr. 8.3 Metoda Talk – připojení metody k postavě

Takto připravený skript umístí veškeré texty ke spodnímu okraji obrazovky a zarovná je na střed bez ohledu na to, kde se právě nachází postava, která text pronáší.

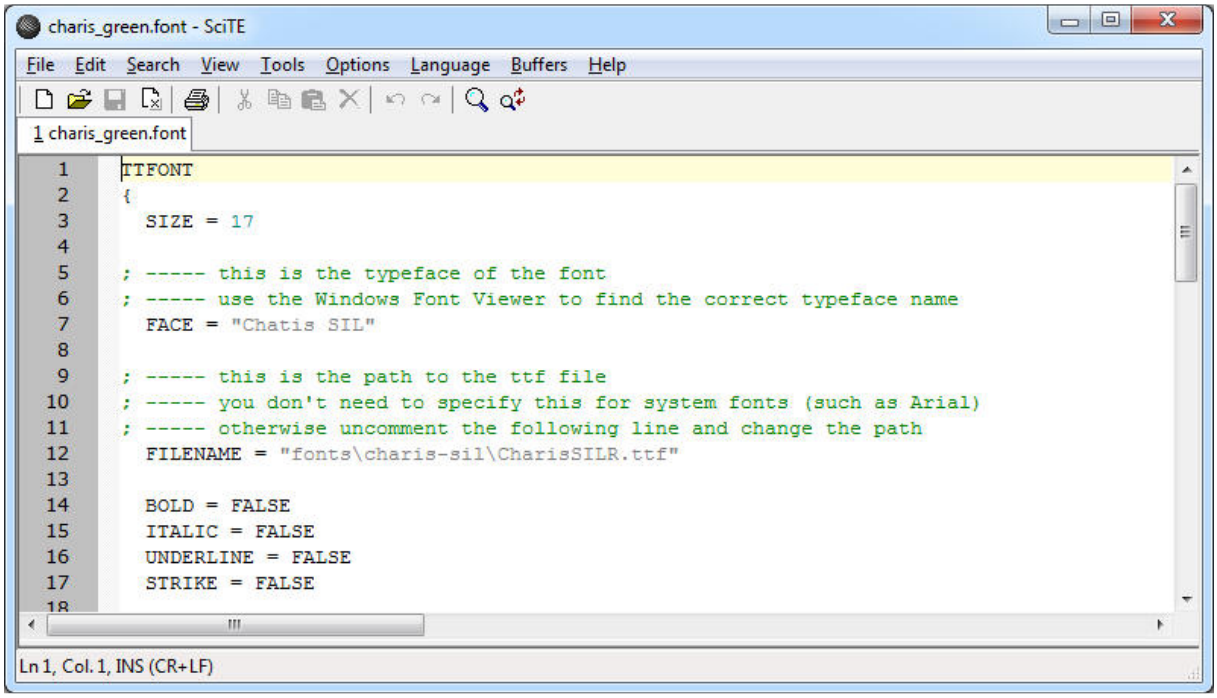
8.2 Přidání vlastního písma ve formátu TrueType

Dalším zmiňovaným problémem souvisejícím s titulky je fakt, že písmo je příliš malé. Ve výchozím stavu obsahuje Wintermute Engine pouze bitmapové fonty. Tyto fonty mají tvary všech písmen uloženy v jediném souboru BMP, odkud jsou načítány potřebné části textury s konkrétními písmeny. S bitmapovými fonty se tedy pracuje jako s texturou. Hlavní výhodou těchto fontů je, že lze poměrně snadno definovat nové znaky, nebo upravovat ty stávající pomocí jednoduchého grafického editoru. Velkou nevýhodou však zůstává, že změna velikosti písma znamená v podstatě úplné předělání zdrojového souboru, kde jsou uloženy tvary a rozměry jednotlivých znaků tohoto písma.
WME však umí pracovat i s vektorovými fonty TrueType. Úprava znaků tohoto písma již není tak snadná, jak tom bylo u znaků předchozího formátu. Oproti bitmapovému formátu má však tento font jednu nespornou výhodu. U fontu TruType lze libovolně měnit velikost písma a písmena jsou ve všech velikostech ostrá a jasně čitelná. Pro použití v této simulaci byl vybrán vektorový font Chatis SIL, který je včetně licence volně stažitelný z internetových stránek www.ceskefonty.cz.
K tomu, aby bylo možné font v programu použít, je nutné pro něj nejdříve ve WME vytvořit konfigurační soubor. Tento soubor vytvoříme kliknutím pravým tlačítkem myši na složku fonts, kde je možné vybrat položku přidat font. Následně je nutné zadat název fontu a vybrat jeho formát. Okno, které se ukáže při vytváření nového konfiguračního souboru fontu, je patrné na Obr. 8.4.



Obr. 8.4 Tvorba konfiguračního souboru fontu

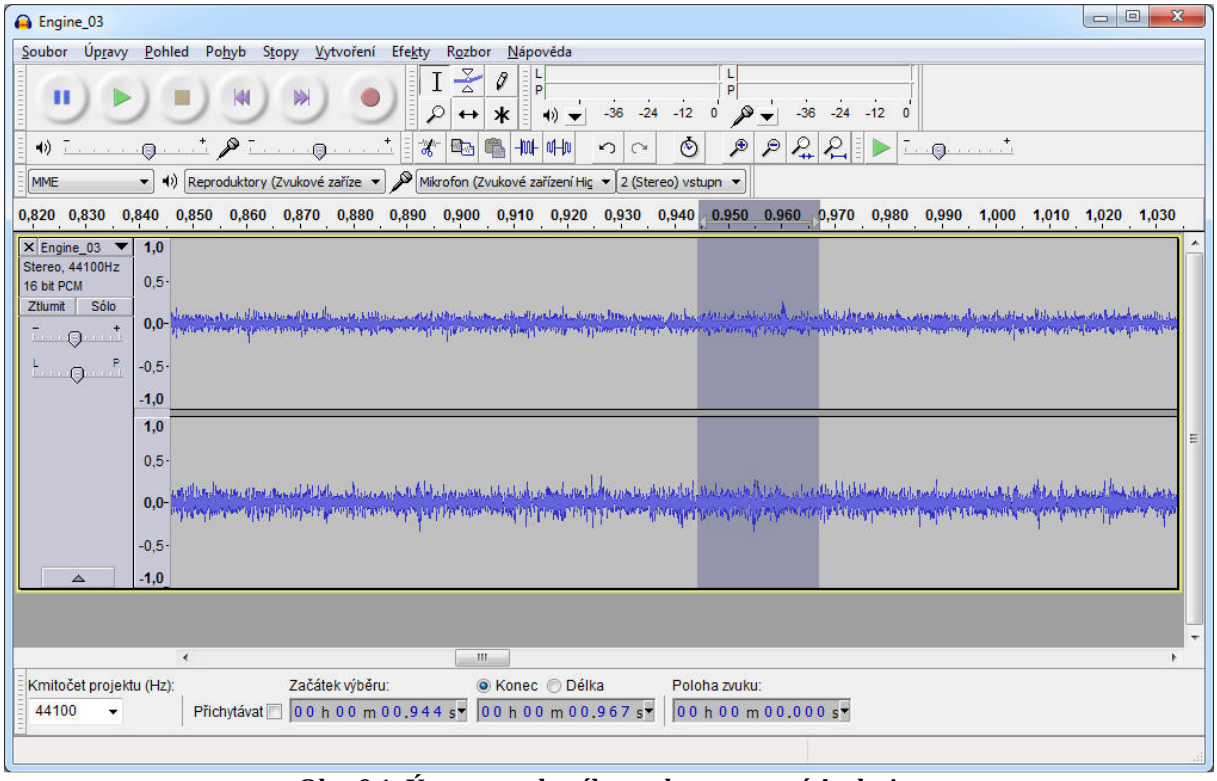
K nově vytvořenému konfiguračnímu souboru (Obr. 8.5) je třeba nejdříve připojit vybraný font. Na řádku 7 je nutné zadat název fontu, který zjistíme ve vlastnostech daného fontu. Řádek 12 pak udává relativní cestu k souboru s fontem. Na řádku 64 na konci tohoto souboru lze také ovlivnit barvu písma zadáním příslušného RGB kódu do položky COLOR. V takto upraveném konfiguračním souboru je nyní možné pomocí parametru na řádku 3 změnit velikost písma na potřebnou hodnotu.



Obr. 8.5 Konfigurační soubor TrueType font

9 Tvorba zvuku

Aby byl vzdělávací efekt předávaných informací co největší, budou veškeré informace o jednotlivých částech stroje studentovi poskytovány kromě textové formy také ve formě mluveného slova. Všechny zvukové nahrávky vytvořil podle dodaných textových podkladů Ing. Petr Hořejší, Ph.D. a poskytl je autorovi.
Nahrávky byly následně pomocí programu Audacity (Obr. 9.1) zesíleny, rozděleny do několika souborů vždy po jedné větě a převedeny do formátu Ogg Vorbis. Audacity je svobodný multiplatformní software, určený pro editaci digitálního zvuku. Aplikace je dostupná pod licencí GPL zdarma na stránkách tvůrce, a to včetně zdrojových kódů. Licence GPL umožňuje volné použití software ke komerčním i nekomerčním účelům, včetně dalšího šíření této aplikace.



Obr. 9.1 Úprava zvukového souboru pomocí Audacity

Takto upravené zvukové soubory byly následně naimportovány do složky sounds umístěné v adresářové struktuře projektu. Poté byl do skriptů jednotlivých objektů předáván ke stávající metodě Talk další parametr, který zajistí přehrání náležitého zvuku v průběhu zobrazení textu.

10 Tvorba virtuálních skript

Základní myšlenka tvorby virtuálních skript byla inspirována virtuálním informačním tabletem, který byl použit v projektu Manažérský simulátor 1 Ing. Miroslava Suchého. Tento tablet však obsahoval některé zásadní nedostatky, proto musel být původní koncept téměř od základu přepracován.

10.1 Informační tablet použitý v projektu MS1

Původní virtuální tablet byl vytvořen jako soustava velkého množství vzájemně propojených oken, které se postupně přes sebe otevíraly. Každé z těchto oken obsahuje jednu stránku s textem a jeden skript, který řídí chod tohoto okna.
Obsah prezentace původního informačního tabletu byl vytvořen pomocí programu MS PowerPoint a následně přes klávesu PrintScreen po jednotlivých listech uložen do programu Malování. Tyto obrázky listů prezentace byly uloženy do jednotlivých oken tabletu. Když uživatel tabletem listuje, otevírání jednotlivých oken způsobí kaskádový efekt, který se projeví při uzavření tabletu probliknutím všech zobrazených stránek. Problém s kaskádovým otevíráním oken způsobuje fakt, že příkazu na řádku 18 Obr. 10.1 na uzavření okna předchází příkaz na řádku 17, který uděluje exkluzivitu nově otevřenému oknu. Zpracovávání tohoto skriptu se tedy pozastaví, dokud se nově otevřené okno neuzavře a tím se uvolní jeho exkluzivita.

1 titulni.script *

```
1  #include "scripts\base.inc"
2  #include "scripts\keys.inc"
3
4
5  ////////////////////////////////////////////////////////////////////////////////
6    on "close"
7  {
8    this.Close();
9  }
10
11 on "Kanban"
12 {
13   // Přepnutí okna na okno Kanban 1
14
15   var k1 = Game.LoadWindow("interface\Tablet\kanban1.window");
16   k1.Center();
17   k1.GoSystemExclusive();
18   this.Close();
19   Game.UnloadObject(k1);
20  }
21
22 on "5whys"
```

Obr. 10.1 Informační tablet [13] – původní skript

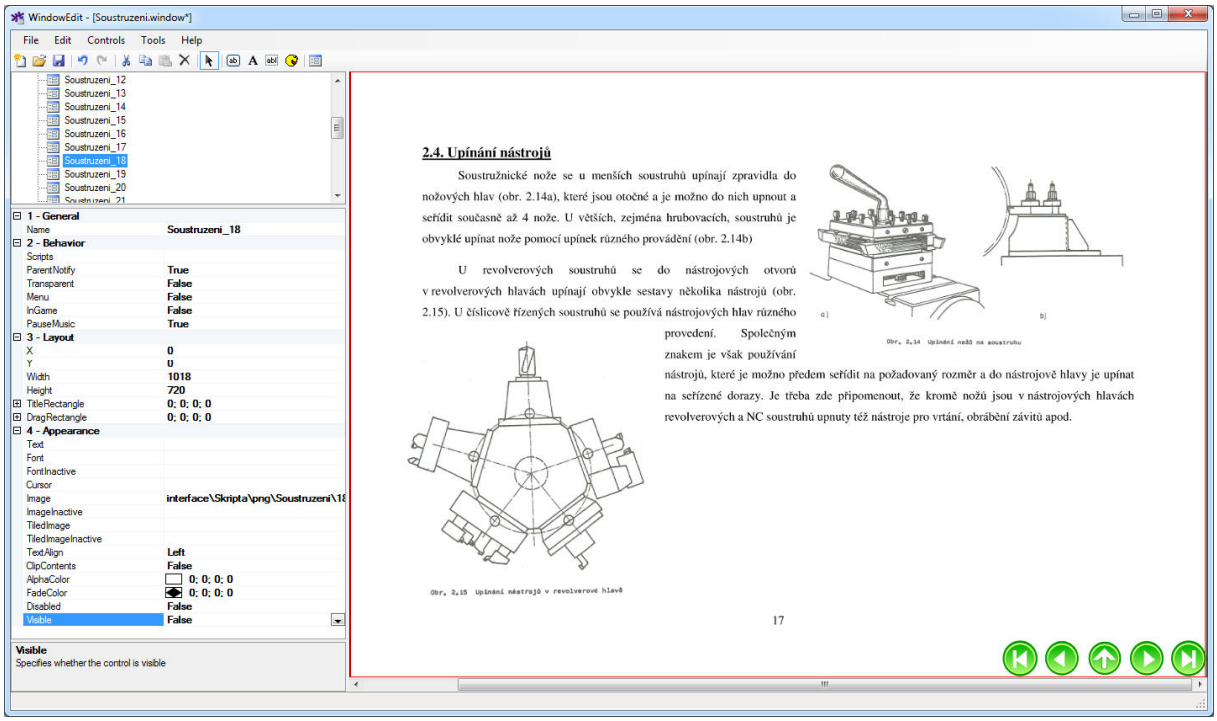
Zde je nutné si uvědomit, že příkaz na řádku 18 není příkaz na uzavření skriptu, ale okna, které skript řídí. Skript tedy dokončí i příkazy, které následují po uzavření okna. Z uvedeného tedy vyplývá, že oprava tohoto problému bude poměrně snadná. Stačí příkaz na uzavření aktuálního okna posunout před udělení exkluzivity nově otevíranému oknu. Opravený skript by mohl vypadat, jak je uvedeno na Obr. 10.2.

Obr. 10.2 Informační tablet – opravený skript

Druhý problém, který způsobuje rozdělení stránek tabletu do několika oken, souvisí s množstvím skriptů potřebných k ovládání tabletu. Každá strana tabletu potřebuje vlastní skript, který obsahuje téměř stejné příkazy, jako skript předcházející stránky. Velké množství oken navíc způsobuje, že je struktura zdrojových souborů projektu nepřehledná a je obtížné pokračovat v dalším vývoji aplikace.

10.2 Virtuální skripta

Virtuální skripta jsou na rozdíl od zmiňovaného informačního tabletu rozdělena pouze do tří oken se studijními texty a jednoho okna řídícího. Každé z těchto tří oken je zaměřeno na jinou metodu obrábění. V řídícím okně si uživatel vybere z obsahu jednu metodu, o které se chce dozvědět více a kliknutím na vybraný řádek otevře příslušné okno, které obsahuje celý studijní text k dané problematice. V tomto okně je možné se libovolně pohybovat po jednotlivých stránkách, nebo skokem na první respektive poslední stranu. Okno lze také z kterékoliv stánky ihned zavřít a vrátit se tak do řídícího okna s obsahem.
Stránky se studijním textem byly vytvořeny v programu MS Word, odkud byly pomocí software PDF Creator, který je opět šířen pod licencí GPL, převedeny na obrázky formátu PNG. Bohužel v tomto programu nelze přesně nastavit grafické rozlišení výstupních souborů, proto musely nově vzniklé obrázky podstoupit ještě jednu transformaci. Tentokrát v programu Gimp byly všechny obrázky znovu přetransformovány na novou velikost 1019x720 bodů, což je výška zobrazovaného grafického rozlišení projektu Virtuální dílna.
Takto upravené stránky skript byly vloženy do příslušného okna a u všech byl nastaven parametr *visible* na hodnotu *false* (Obr. 10.3). Jako poslední prvky těchto oken byla vložena navigační tlačítka, proto budou tato tlačítka vždy viditelná a nepřekryje je žádný list se studijním textem. Tlačítko pro zobrazení další strany je nazváno next, tlačítko pro předchozí stranu prev, tlačítka pro přesun na první, nebo poslední stranu mají název first respektive last a konečně tlačítko pro zavření aktuální kapitoly je nazváno close.



Obr. 10.3 Window editor – virtuální skripta

Funkce tlačítek je definována v příslušném skriptu. Na Obr. 10.4 jsou patrné příkazy, zajišťující pohyb po jednotlivých stránkách vpřed a vzad. Po kliknutí na tlačítko next, zkontroluje skript nejdříve pomocí podmínky na řádce 21, jestli se již nenachází na poslední straně.

1 Soustruzeni.script

```
17  // posun na další stranu
18  ////////////////////////////////////////////////////////////////////////////////
19  on "next"
20  {
21      if (Strana < PocetStran)
22      {
23          Strana = Strana + 1;
24          Zobraz = this.GetControl("Soustruzeni_" + ToString(Strana));
25          Zobraz.Visible = true;
26      }
27  }
28
29  // posun na předchozí stranu
30  ////////////////////////////////////////////////////////////////////////////////
31  on "prev"
32  {
33      if (Strana > 1)
34      {
35          Zobraz = this.GetControl("Soustruzeni_" + ToString(Strana));
36          Zobraz.Visible = false;
37          Strana = Strana - 1;
38      }
39  }
```

Obr. 10.4 Virtuální skripta – změna stránky pomocí myši

Jestliže tomu tak není, přičte následující příkaz hodnotu 1 k proměnné Strana, která uchovává číslo aktuálně zobrazené strany. Na řádce 24 načte do proměnné Zobraz tuto aktuální stranu a následujícím řádkem změní její parametr *visible* na hodnotu *true*, čímž dojde k zobrazení

strany. Událost stlačení tlačítka prev je ošetřena na řádcích 31 až 39. Zde podmínka na řádce 33 nejdříve prověří, jestli není zobrazena první strana. Následně skript přepne na aktuální straně parametr *visible* na hodnotu *false* a poté příkaz následujícím řádku upraví hodnotu proměnné uchovávající číslo aktuálně zobrazené strany.

1 Soustruzeni.script

```
83  // funkce kláves
84  ////////////////////////////////////////////////////////////////////////////////
85  on "Keypress"
86  {
87      var button;
88
89      if(Keyboard.KeyCode==VK_ESCAPE)
90      {
91          button = this.GetControl("close");
92          button.Press();
93      }
94      else if(Keyboard.KeyCode==VK_RIGHT)
95      {
96          button = this.GetControl("next");
97          button.Press();
98      }
99      else if(Keyboard.KeyCode==VK_END)
100     {
101         button = this.GetControl("last");
102         button.Press();
103     }
104     else if(Keyboard.KeyCode==VK_LEFT)
105     {
```

Obr. 10.5 Virtuální skripta – ovládání z klávesnice

Ovládání virtuálních skript pomocí kláves řeší událost Keypress, uložená na konci téhož skriptu (Obr. 10.5). Tato událost vyvolá po stisknutí definované klávesy příslušnou proceduru popisovanou v předcházejícím textu, která zajistí požadovanou změnu zobrazené strany.

11 Uživatelský manuál

Přestože během tvorby programu byl kladen velký důraz na jednoduché a intuitivní ovládání, nemusí být každému uživateli okamžitě jasné, jak program ovládat. Proto bude vhodné nyní uvést několik informací o ovládání programu.

11.1 Ovládání

Program Virtuální dílna se ovládá pomocí myši v kombinaci s několika funkčními klávesami.

Levé tlačítko myši – zběžné prozkoumání objektu, sebrání předmětu, použití předmětu, změna scény.

Pravé tlačítko myši – podrobnější prozkoumání objektu, prozkoumání/použití předmětu v inventáři.

Klávesa Esc – zobrazení hlavního menu, opuštění detailního náhledu na prozkoumávaný předmět.

Klávesa F1 – zobrazení nápovědy.

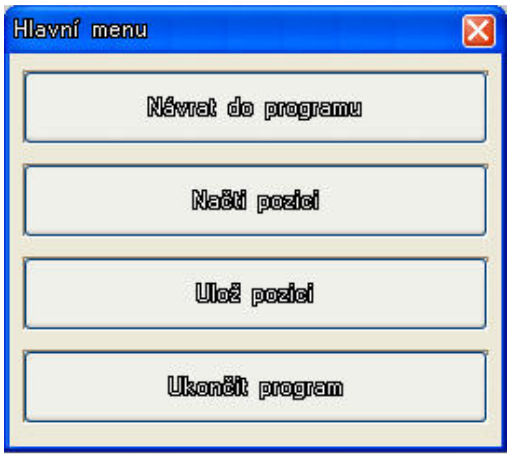
11.2 Menu

Návrat do programu – uzavřít menu a návrat zpět do simulace.

Načíst pozici – opustit současnou simulaci a pokračovat v dříve uložené simulaci.

Uložit pozici – uložit současný stav simulace.

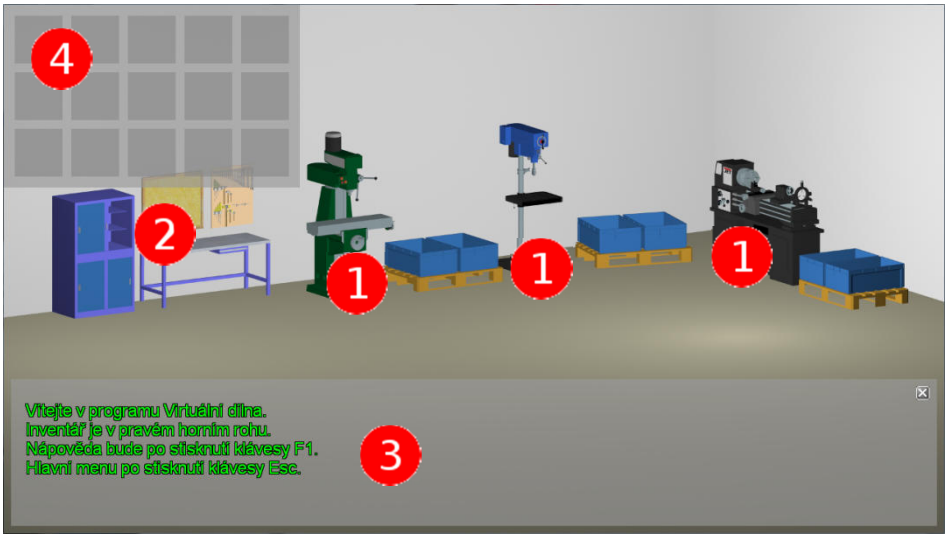
Ukončit program – opustit současnou simulaci a návrat do operačního systému. Neuložená simulace bude ztracena.



Obr. 11.1 Hlavní menu

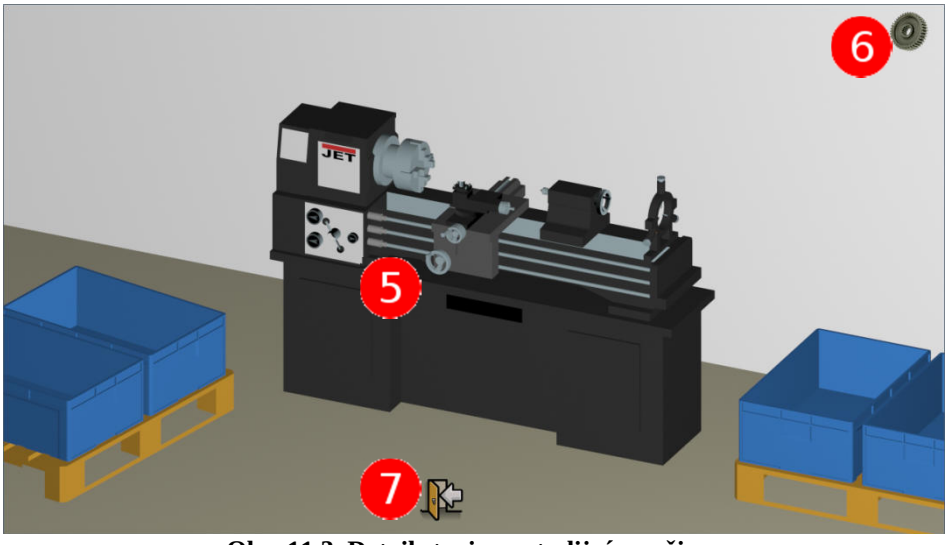
11.3 Simulace

Simulace je rozdělena do několika obrazovek. Hlavní obrazovka je celkový pohled do dílny, odkud lze po kliknutí levým tlačítkem myši přistupovat k jednotlivým strojům (1). Zvláštní scénou je pracovní stůl s nástěnkou (2). Zde je možné nalézt virtuální skripta a tato obrazovka zároveň slouží k zadávání úkolů.
Ve spodní části obrazovky je zobrazeno okno s několika informacemi, které se mohou do začátku hodit (3). Toto okno se uzavírá křížkem v pravém horním rohu. Po uzavření tohoto okna ho již nelze až do opětovného spuštění aplikace znova otevřít.
V levém horním rohu se nachází inventář (4), který se zpřístupní po najetí kurzorem myši do tohoto rohu. Zde jsou uloženy předměty, které získáte během simulace.



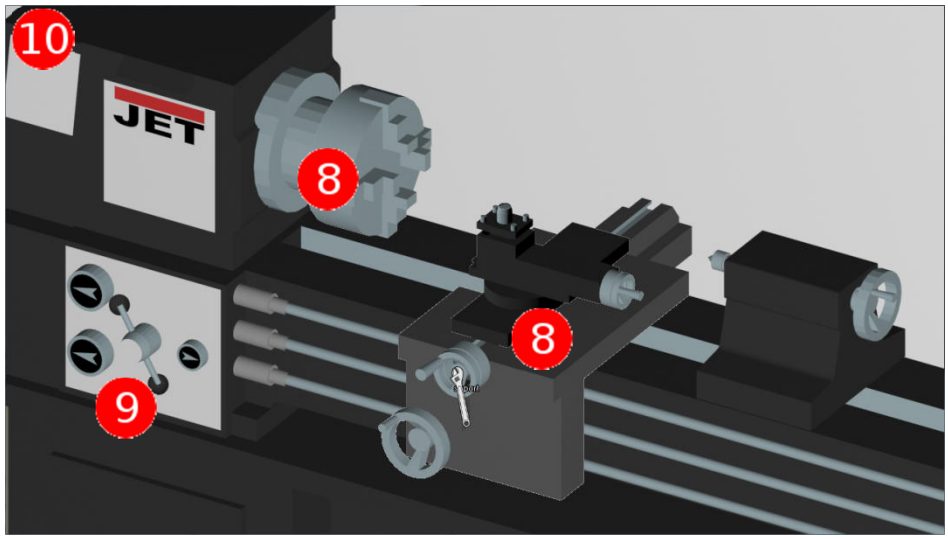
Obr. 11.2 Celkový pohled do dílny

Detail každého stroje je rozdělen do dvou režimů. Ve studijním režimu lze pomocí obou tlačítek myši prozkoumávat jednotlivé části stroje (5). Změna do praktického režimu je možná pomocí tlačítka s ozubeným kolem v pravém horním rohu obrazovky (6). Toto tlačítko se aktivuje až po splnění úkolu na scéně s nástěnkou. Návrat do hlavního náhledu dílny je umožněn po najetí kurzoru myši do spodní části obrazovky, kde se kurzor změní na symbol dveře (7).



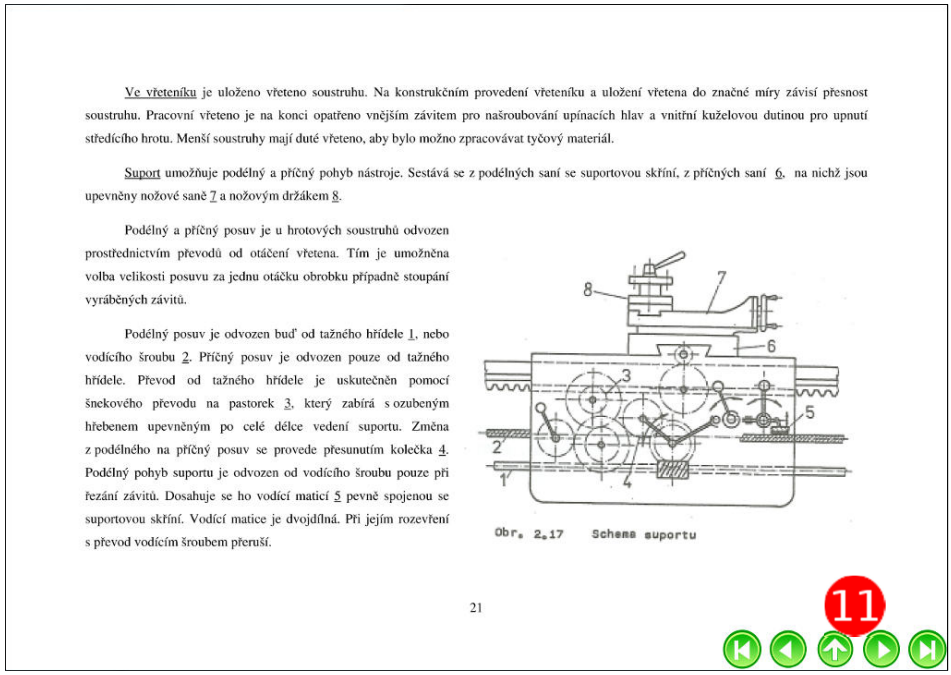
Obr. 11.3 Detail stroje ve studijním režimu

Druhý režim virtuální praxe je přístupný pomocí příslušného tlačítka (6) ze studijního režimu každého stroje. Části stroje v tomto režimu již nejsou opatřeny podrobnými popisy, zato lze aktivní části stroje (8) ovládat pomocí příslušných ovládacích prvků (9). Inventář (4) je samozřejmě dostupný i v tomto režimu v levém horním rohu (10).



Obr. 11.4 Režim virtuální praxe

Po celou dobu simulace jsou dostupná virtuální skripta s podrobnými informacemi o problematice zde přítomných strojů. Listování ve skriptech je možné pomocí navigačních tlačítek (11) v pravé spodní části obrazovky, nebo šipek na klávesnici. Ukončení prohlížení virtuálních skript je možné buď s využitím příslušného navigačního tlačítka (11), nebo pomocí klávesy Esc.



Obr. 11.5 Virtuální skripta

12 Závěr

Cílem této práce bylo vytvořit výukový program z prostředí výrobního podniku, který zábavnou formou seznámí uživatele s několika stroji přítomnými ve virtuální dílně. Při tvorbě byl využit softwarový balík Wintermute Engine, který se vyznačuje především svojí otevřeností a flexibilitou. Tato skutečnost dává na jednu stranu programátorovi možnost uzpůsobit si všechny přednastavené šablony objektů a nástroje pro tvorbu vlastním potřebám, ale na druhou stranu není vždy snadné dosáhnout požadovaného cíle. Některých záměrů bylo možné dosáhnout pouze za cenu přímé editace kódu výchozích metod a funkcí balíku.
V rámci projektu Virtuální dílna byl vytvořen model dílny, do kterého byly zasazeny předváděné stroje. Každý z těchto strojů byl v programu následně podrobně popsán a vysvětleny hlavní principy jeho fungování. Další významnou částí programu jsou virtuální skripta, kde se uživatel může dozvědět další užitečné informace. Poslední částí simulace je proces výroby jednoduchého dílu podle přítomné dokumentace. Očekávaný efekt této poslední části programu je spíše motivační, proto není obsluha strojů a výroba zmiňovaného dílu zachycena s naprostou věrností v porovnání se skutečnou výrobou obdobných výrobků. Hlavním účelem tohoto programu je sloužit jako studijní pomůcka pro studenty prvních ročníků fakulty strojní.
Současný trend vývoje lidské společnosti způsobuje odklon mládeže od všech technických oborů a jejich zvýšený zájem o humanitní vědy. Přitom je ale nutné si uvědomit, že výrobní podnik je jediným skutečným místem vzniku hospodářské a ekonomické hodnoty. Proto je nezbytné technické obory mezi mládeží neustále propagovat. K této propagaci bude mimo svého hlavního účelu přispívat na základních a středních školách v rámci projektu Populár i tento program.

13 Literatura

[1] BUREŠ, M., LEEDER, E., *Digitální podnik*, [e-book] Plzeň : ZČU, 2012.

[2] HOŘEJŠÍ, P., GÖRNER, T. a KURKIN, O, *VYZTYMDP : Virtuální realita : základní úroveň*, [e-book] Plzeň : ZČU, 2012. ISBN 978-80-87539-07.

[3] „cs.wikipedia.org," retrieved 25. 11 2013, http://cs.wikipedia.org/wiki/Simulace.

[4] GÖRNER, T., HOŘEJŠÍ, P. a KURKIN, O., *VYZTYMDP : Virtuální realita : úvodní úroveň*, [e-book] Plzeň : ZČU, 2012. ISBN-978-80-87539-07.

[5] „en.wikipedia.org," retrieved 25. 11 2013, http://en.wikipedia.org/wiki/Serious_game.

[6] „http://dead-code.org," retrieved 22. 11 2013, http://dead-code.org/home/index.php/about/.

[7] „en.wikipedia.org," retrieved 22. 11 2013, http://en.wikipedia.org/wiki/Wintermute_Engine.

[8] „cs.wikipedia.org," retrieved 22. 11 2013, http://cs.wikipedia.org/wiki/Paralaxní_scrolling.

[9] „www.scintilla.org," retrieved 26. 11 2013, http://www.scintilla.org/index.html.

[10] „en.wikipedia.org," retrieved 25. 11 2013, http://en.wikipedia.org/wiki/SciTE.

[11] „res.dead-code.org," retrieved 29. 11 2013, http://res.dead-code.org/doku.php/games:start.

[12] „www.juliathegame.com," retrieved 25. 11 2013, http://www.juliathegame.com/.

[13] SUCHÝ, M., *Tvorba vážné hry v konceptu digitální továrny*, [e-book] Plzeň : ZČU, 2013.

[14] BURDEA, G. a GRIGORE, C., *Virtual Reality Technologi, Second Edition*, místo neznámé : Wiley-IEEE Press, 2003. ISBN 0-471-36089-9.

[15] SOVA, F., *Technologie obrábění a montáže*, Plzeň : ZČU, 2001. ISBN 80-7082-823-4.