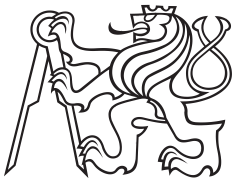
Bakalářská práce



České
vysoké
učení technické
v Praze

F3

Fakulta elektrotechnická
Katedra počítačové grafiky a interakce

Bakalářská práce

Kompletace 3D modelu Pražského hradu

Jiří Ryšavý

Vedoucí: Prof. Ing. Jiří Žára, CSc.
Obor: Počítačové hry a grafika 2016
Studijní program: Otevřená informatika
Duben 2020

Poděkování

Chtěl bych poděkovat všem, kteří mě podporovali během studia. Nejvíce pak své rodině, přátelům a kolegům. Také bych rád poděkoval svému zaměstnavateli za umožnění práce při studiu a úpravu pracovních podmínek dle mých potřeb.

Prohlášení

Prohlašuji, že jsem předloženou práci vypracoval samostatně, a že jsem uvedl veškerou použitou literaturu.

V Praze, 29. dubna 2020

Abstrakt

Tato práce se zaobírá návrhem, implementací a testováním uživatelského editoru pro online model Pražského hradu. Díky tomuto nástroji je možné upravovat jednotlivé části, přidávat nové části a odebírat části z aktuálního modelu. Celý nástroj je napsán v jazyce JAVA s knihovnami pro FTP komunikaci a zpracovávání JSON souborů.

Klíčová slova: COLLADA, Editor, Objekt, Model

Vedoucí: Prof. Ing. Jiří Žára, CSc.
Praha 2, Karlovo náměstí 13, E-320

Abstract

This work deals with design, implementation and user testing of editor for online model of prague castle. This tool can change, add and remove parts from actual model. The tool is written in JAVA language and uses libraries for FTP connection and work with JSON files.

Keywords: COLLADA, Editor, Object, Model

Title translation: Bachelor thesis — Completion of Prague Castle 3D model

Obsah

Obrázky

Kapitola 1

Úvod

Model Pražského hradu je původně webová aplikace vytvořená studenty Antonínem Smrčkem [1] a Janem Husákem[2]. Na základě komunikace s jedním z autorů webového modelu Pražského hradu vyšlo najevo, že Pražský hrad nemá lehkou editovatelnost. Je tedy velice vhodné tento nástroj vytvořit, abychom mohli model Pražského hradu vylepšovat i v následujících letech. Samotný editor nám také dává možnost přidávat do modelu části, se kterými původně nebylo počítáno. Dále také umožňuje oddělení programátorské činnosti od uživatelské, kde dříve pro editaci byla nutná znalost samotné webové aplikace.
Cílem práce je navrhnout a implementovat nástroj pro editaci online modelu Pražského hradu v jazyce JAVA. Editor uživateli umožní jednoduchou manipulaci s objekty již v modelu umístěnými, ať už jejich úpravou, odstraňováním či přidáváním. Zobrazí uživateli informace o objektech v modelu umístěných. Dále uživatelům umožní přidat k objektům informačí popisky a procházkový bod.
Dalším bodem práce je vytvoření uživatelské příručky, která popíše, jak se s daným editorem manipuluje. Editor následně necháme otestovat uživateli a zaznamenáme uživatelské připomínky.
Do teoretické části popíšeme potřebné znalosti pro implementaci editoru a rozbor souborů používaných ve webové aplikaci. Dále pak popíšeme knihovny použité v aplikaci. Následně se přesuneme k implementaci, kde popíšeme práci s jednotlivými částmi, uvedeme ukázky práce s knihovnami a utvoříme návrh uživatelského rozhraní. V závěru práce pak necháme uživatele otestovat aplikaci podle přiděleného scénáře. Dle testování také přibylo několik návrhů pro zlepšení editoru, které zde také shrneme.

Kapitola 2

Model Pražského hradu

Aplikace je psána v javascriptu. Nás zajímají hlavně části obsahující informace o modelu, cestě, kolizích a informačních panelech. Pro nalezení částí bylo nutné podrobně prostudovat strukturu kódu, funkčnost jednotlivých skriptů a návazností při načítání aplikace.

2.1 Použité soubory

Projitím kódu jsme nalezli potřebné informace pro jednotlivé části. Samotný model s pomocí souboru typu COLLADA(.dae). Přesná pozice je: `web_root/data/scene/prague_castle.dae`. Dalším bodem je schopnost automatické procházky po mapě s možností zvolení zajímavých míst. Funkce procházky je vlastně vyexportovanou křivkou do grafu a uloženého v souboru JSON. Cesta k JSON souboru je: `web_root/data/tour/tour.json`. Poslední důležitou částí jsou popisující informační panely pro zvýrazněné části modelu. Všechny popisky jsou uloženy v jednotlivých souborech HTML. Adresář s popisky se nachází na: `web_root/info/`. V adresáři je i umístěn vzorový soubor, podle kterého se jednotlivé popisky vytváří. Soubor se se jmenuje `_template.html`. Kolizní systém je v původní aplikaci řešen pomocí heightmap pro tři stupně podlaží. Heightmapy jsou umístěny v adresáři `web_root/data/heightmap/`. V našem případě ovšem potřebujeme heightmapy generovat při každé změně modelu. Autoři původní aplikace s něčím podobným již počítali a proto stačí pouze přepnout parametr *GENERATE_HEIGHTMAP* z `false` na `true` v souboru `web_root/src/app/Preferences.js`. Nyní si řekneme něco o použitých souborech.

2.1.1 JSON

[3]V tomto souboru lze uložit objektovou strukturu pro libovolnou třídu JavaScriptu. Syntaxe tohoto systému je celkem snadná. Veškeré informace jsou uloženy ve formě objektů `{klíč : hodnota}`, kde klíč je buď string, nebo int. Hodnota je pak pole, číslo, string, nebo další objekt. Do JSONu je uložena procházka. Pro procházku máme vytvořené tři klíče. Klíče jsou *"texts", "vertices", "indicies"*. Pod klíčem *texts* je umístěn objekt. Tento objekt obsahuje pouze body, které mají na mapě přidělený procházkový text. Pod klíči jsou uloženy texty a v hodnotách jsou umístěny inexy bodů, u kterých dané texty jsou. *Vertices* má jako hodnotu pole, ve kterém jsou umístěny globální x, y, z souřadnice všech bodů. Uložené jsou postupně jako $[x_0, y_0, z_0, x_1, y_1, z_1, ..., x_n, y_n, z_n]$, kde n je počet bodů cesty bez jedné. Nakonec *indicies* obsahuje pole po sobě jdoucích indexů bodů. Tyto indexy po dvojicích určují hrany. Vypadá následovně: $[e_{0_p}, e_{0_k}, e_{1_p}, e_{1_k}, ..., e_{m_p}, e_{m_k}]$, kde m je počet hran grafu, p značí počátek hrany a k její konec. V tomto případě používáme označení počátku a konce pouze pro lepší pochopení uložení v souboru. Na funkčnost nemají počátky a konce vliv, jelikož v implementaci se graf považuje za neorientovaný.

2.1.2 COLLADA

COLLADA[4] je způsob, jakým lze ukládat všechny informace o modelu ve formě XML. Tedy lze ukládat informace mezi dvojice tagů následovně:

```
<tag1>informace1</tag1>
<tag2>
    <tag3>informace2</tag3>
    <tag4/>
</tag2>
```

V našem případě používáme COLLADA verzi 1.4.1. To nám umožňuje ukládat objekty, materiálové efekty, UV souřadnice a závislosti mezi jednotlivými částmi modelu. Model je uložen jako soubor trojúhelníků.
Nyní si probereme jednotlivé tagy, které jsou pro naši následnou implementaci důležité. Tag `asset` obsahuje informace o souboru a globální nastavení světa, ve kterém je model umístěn. `library_images` obsahuje všechny použité obrázky v modelu. Ať už se jedná o textury či normálové mapy. `library_effects` definuje efekty, neboli osvětlovací modely použité u objektů. Dalším tagem v řadě je `library_materials` kde se pouze materiálům přiřazují efekty z `library_effect`. Nyní následuje asi nejdůležitější tag a to `library_geometries`, ve kterém jsou uložené jednotlivé meshe. Mesh se

ukládá ve formě pole s jednotlivými body. Díky tomu, že celý model je uložen ve formě trojúhelníků, můžeme hrany definovat jako pole indexů, kde se po třech čtou jednotlivé trojúhelníky. Index je trojice umístěná v poli s body, tedy index nula v poli indexů je trojice nula, jedna, dva, v poli bodů. Tato část nese nejvíce datových informací a proto také je nejrozsáhlejší. `library_geometries` definuje body v jejich lokálním souřadném systému, následuje část, ve které přidělíme jednotlivým bodům jejich globální umístění. Část je uložena pod tagem `library_visual_scenes`. V této části je pro každou mesh, definovanou v předchozí části, uložena transformační matice, která objektu přidělí pozici v globálním souřadném systému. Posledním použitým tagem je `scene`, kde se pouze přidělí použitá scéna. V modelu Pražského hradu existuje pouze jedna scéna, tedy ta se přidělí.

2.1.3 HTML

Obdobně jako COLLADA se jedná o program zapsaný v XML formátu. Jedná se o běžně používaný jazyk pro zobrazování informací na webech.[5] V současné době je nejnovější HTML verze 5. To, jak ho v našem případě použijeme my, však může být napsáno i v nižších verzích. Použijeme tagy `a, article, div, h1, img, li` a `ul`. Tag `a` využijeme jako odkazy na stránky zdrojů pro informační panel. `Article` máme jen pro určení obsahu pro panel. Stejně tak `div` užívámě jako oddělovač jednotlivých částí. `H1` zde máme pro nadpis informačního panelu. Pro zobrazení obrázku na panelu využijeme `img` tag. Seznamové tagy `ul` a `li` používáme protože odkazů zdrojů může být více. Tedy tagy `a` ukádáme do seznamů `ul` pod jednotlivé `li`.

2.2 Použité knihovny

V aplikaci použijeme některé knihovny pro zjednodušení implementace našeho vlastního kódu.

2.2.1 JAMA

Jama je knihovna pro základní operace z lineární algebry[6]. Umožňuje nám jak lehčí operace, jako sčítání, odčítání matic, tak složitější, jako je třeba

násobení. Knihovna dodává i náročnější operace, jako je QR rozklad a podobné. V naší implementaci využijeme však pouze násobení a sčítání matic.

2.2.2 Commons-net

Commons-net je knihovna umožňující síťovou komunikaci s pomocí nejběžnějších protokolů. Výběrem například SMTP, Telnet, HTTP, SNTP a FTP[7]. My tuto knihovnu použijeme pro FTP komunikci s FTP serverem, který nám umožňuje správu souborů na webu.

2.2.3 Json simple

Json simple, je knihovna umožňující čtení a zápis JSON souborů pro jazyk Java. Umožňuje načítat obsah JSON souborů do polí, nebo do objektů[8]. Využijeme tuto knihovnu pro dekódování obsahu souborů s cestou.

Kapitola 3

Implementace

Implementaci rozvrhneme dle návrhového vzoru MVC do částí model, view a controller. Model reprezentuje funkční datovou část programu. V této části budeme zpracovávat jednotlivé soubory. Budeme zde též řídit komunikaci mezi FTP serverem a naší aplikací. A také tu budeme zjišťovat výpočty potřebné pro naši aplikaci. Část view je využívána pro interakci s uživatelem. Vesměs se jedná pouze o uživatelská rozhraní. Poslední částí je pak controller, která zařizuje komunikaci mezi částí view a model. V kódu tedy obsahuje odchycení signálu od uživatele, jeho zpracování a zaslání informace do modelu. Jelikož část controller není z hlediska implementace nijak složitá, obsahuje pouze posluchače tlačítek a získává informace potřebné pro model, podrobnější popis v textu vynecháme.

3.1 Model

Při načítání aplikace stáhneme ze serveru konfigurační soubor a soubor s modelem. Z konfiguračního souboru načteme veškeré informace a spustíme aplikaci. V případě, kdy uživatel otevře okno pro úpravu textu, stáhne se navíc i soubor informačního panelu.

3.1.1 Operace se soubory

Jelikož každý objekt v modelu má určité parametry, které je třeba znát, aby se s nimi správně pracovalo, vytvoříme konfigurační soubor. Konfigurační soubor bude v textovém formátu, kde budeme uvádět parametry a oddělovat je budeme symbolem "/". Konfigurační soubor necháme obsáhnout následující informace: Název objektu, hlavní uzel objektu v souboru COLLADA, informační panel ano/ne, index bodu procházky, rotace kolem osy X, rotace kolem osy Y, rotace kolem osy Z, posun po ose X, posun po ose Y, posun po ose Z, zvětšení po ose X, zvětšení po ose Y, zvětšení po ose Z, bod procházky X souřadnice, bod procházky Y souřadnice, bod procházky Z souřadnice, identifikátory v souboru COLLADA. Každý řádek souboru reprezentuje jeden objekt. Všechny informace si načteme.
Při přidávání objektu do modelu procházíme postupně přes tagy uvedené v části o COLLADA souboru. Procházíme jak původní soubor, tak soubor přidávaného objektu. Vždy přepíšeme nejdřív data z původního souboru, následně pak z nového. Oba soubory takto postupně spojujeme do nového souboru, kterým po spojení nahradíme původní soubor modelu. Během přidávání si zaznamenáváme identifikátory nového objektu, které následně přidáme i do konfiguračního souboru. Při procházení si zaznamenáme i názvy obrázkových souborů, které objekt používá. Obrázkové soubory se při exportu objektu ukládají do stejného adresáře, jako soubor s objektem. Tyto soubory při odesílání soubrů na server odešleme spolu s novým souborem modelu. Pro vizuální představu je zde ukázka 3.1.
Pokud objekt odebíráme, procházíme postupně identifikátory z konfiguračního souboru v souboru s modelem a příslušné tagy přeskočíme. Vizuálně se stane následující 3.2.
Záměna objektu je pak kombinací předchozích dvou, kdy si při odstranění zapamatujeme transformační matici a nově přidanému objektu ji přiřadíme.
Přidávání informačního panelu je triviální. HTML soubor má danou strukturu. Tedy části zapíšeme přesně dle vzoru do nového souboru. Propojení mezi informačním panelem a samotným objektem řeší soubor `web_root/src/strings/ContentLibrary.js` při přidávání informačního panelu tedy stáhneme i tento soubor a na jeho konec přidáme řádku ve formátu:

```
1    ContentLibrary.createContentString("main_node", "text_EN" ,
    "text_CZ");
```

Listing 3.1: Propojení informačního panelu a objektu

kde main_node je hlavní uzel objektu, text_EN je text zobrazený při najetí myší v anglickém jazyce a text_CZ má stejnou funkci, jen pro český jazyk.
Poslední částí je pak vkládání do práce se soubory JSON. K tomuto využíváme knihovnu Json simple. Při načítání umí knihovna dekódovat do ArrayMap s pomocí iterátorů. Předáme knihovně klíč, který chceme z objektu vytáhnout

a projdeme vrácené pole hodnot. Při zápisu naopak ArrayMapu zapíšeme pod daný klíč.

3.1.2 Komunikace se serverem

Pro komunikaci se serverem využíváme knihovnu Commons-net. Připojujeme se s pomocí FTP protokolu. Pro komunikaci je třeba znát adresu serveru, port komunikace, uživatelské jméno a heslo. Dále je třeba znát cestu k souboru serverové části a cestu k souboru lokální části. Soubor přesouváme jako stream. Tedy postupně data přeléváme ze vstupního souboru do výstupního.

```
1   public void downloadFromServer() {
2       //class from library
3       FTPClient client = new FTPClient();
4
5       try {
6           client.connect(FTP_SERVER, PORT);
7           client.login(USERNAME, PASSWORD);
8
9           File localfile = new File(localFile);
10
11          OutputStream outputStream = new BufferedOutputStream
    (new FileOutputStream(localfile));
12          client.retrieveFile(serverFile, outputStream);
13          outputStream.close();
14
15      }
16      catch (IOException ex) {
17          //"Error occurs in downloading files from ftp Server
    "
18      }
19  }
```

Listing 3.2: Stažení textového souboru z FTP serveru

3.1.3 Výpočty v aplikaci

V aplikaci provádíme výpočet transformační matice ze zadaných parametrů a výpočet nejkratší cesty mezi dvěmi body.
Transformační matice je definována jako: $T * R * S * O$, kde T je matice posunu, R je rotační matice S je matice zvětšení a O je počátek, v našem případě O je jednotková matice.

T=$\begin{bmatrix} 1 & 0 & 0 & t_x \\ 0 & 1 & 0 & t_y \\ 0 & 0 & 1 & t_z \\ 0 & 0 & 0 & 1 \end{bmatrix}$, kde t_x, t_y, t_z reprezentují posun v jednotlivých osách.

S=$\begin{bmatrix} s_x & 0 & 0 & 0 \\ 0 & s_y & 0 & 0 \\ 0 & 0 & s_z & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$, kde s_x, s_y, s_z reprezentují zvětšení v jednotlivých osách.

Matice rotace je o něco složitější. $R = R_x * R_y * R_z$, kde

$$R_x = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & cos(r_x) & -sin(r_x) & 0 \\ 0 & sin(r_x) & cos(r_x) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$R_y = \begin{bmatrix} cos(r_y) & 0 & sin(r_y) & 0 \\ 0 & 1 & 0 & 0 \\ -sin(r_y) & 0 & cos(r_y) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$R_z = \begin{bmatrix} cos(r_z) & -sin(r_z) & 0 & 0 \\ sin(r_z) & cos(r_z) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Zde r_x, r_y, r_z reprezentují rotace kolem příslušných os.
Hodnoty $t_x, t_y, t_z, s_x, s_y, s_z, r_x, r_y, r_z$ vyčteme z políček, která nám dal uživatel. K násobení matic používáme knihovnu JAMA, které se matice zadají jako dvourozměrné pole. Násobení matic pak řeší funkce times.

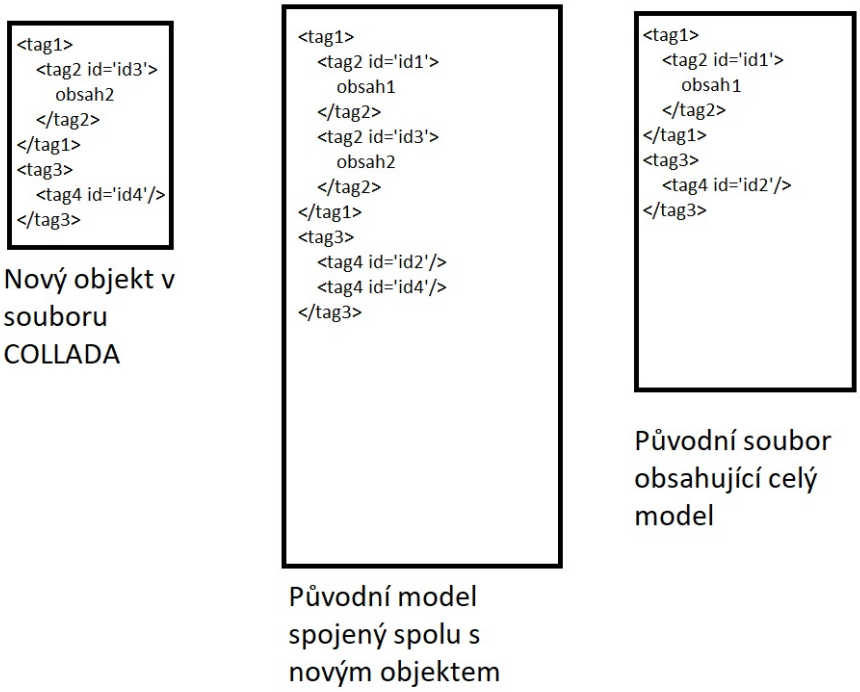
```
Matrix T = new Matrix(
    new double[][]{
        {1.0, 0.0, 0.0, 1.0},
        {0.0, 1.0, 0.0, 0.0},
        {0.0, 0.0, 1.0, 0.0},
        {0.0, 0.0, 0.0, 1.0}
    });
Matrix S = new Matrix(
    new double[][]{
        {2.0, 0.0, 0.0, 0.0},
        {0.0, 2.0, 0.0, 0.0},
        {0.0, 0.0, 2.0, 0.0},
        {0.0, 0.0, 0.0, 1.0}
    });
Matrix R = new Matrix(
    new double[][]{
        {1.0, 0.0, 0.0, 0.0},
        {0.0, 0.0, -1.0, 0.0},
        {0.0, 1.0, 0.0, 0.0},
        {0.0, 0.0, 0.0, 1.0}
    });
Matrix result = (T.times(R)).times(S);
```

Listing 3.3: Použití knihovny JAMA pro násobení matic posunu o 1 po ose x

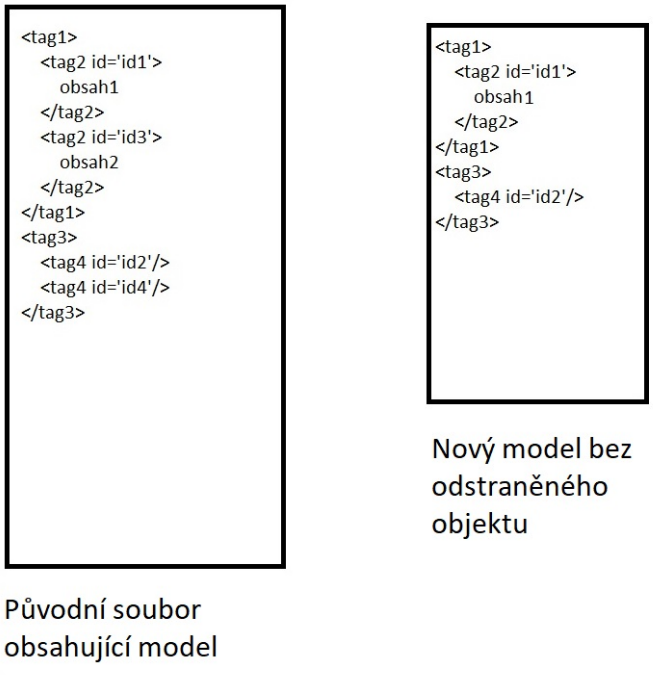
Výslednou transformační matici poté posíláme do funkce pro přidávání objektů a v části, kde se řeší přenesení objektu do globálních souřadnic, matici vložíme mezi tagy `matrix`. Důležité je dbát na to, aby se matice vložila pouze nejnadřazenějšímu uzlu objektu, pokud je objekt složen z více částí.
Hledání nejkratší vzdálenosti mezi uzly řešíme pouze v případě, kdy chce uživatel posunout bod procházky, nebo ho chce přidat. Hledáme s pomocí Pythagorovy věty nejkratší hranu mezi bodem a všemi body již v grafu umístěnými. V případě, kdy hranu nalezneme, zapamatujeme si index uzlu pro napojení. Poté do grafu přidáme novou dvojici tvořící hranu mezi novým bodem a bodem s nejkratší hranou.

3.2 View

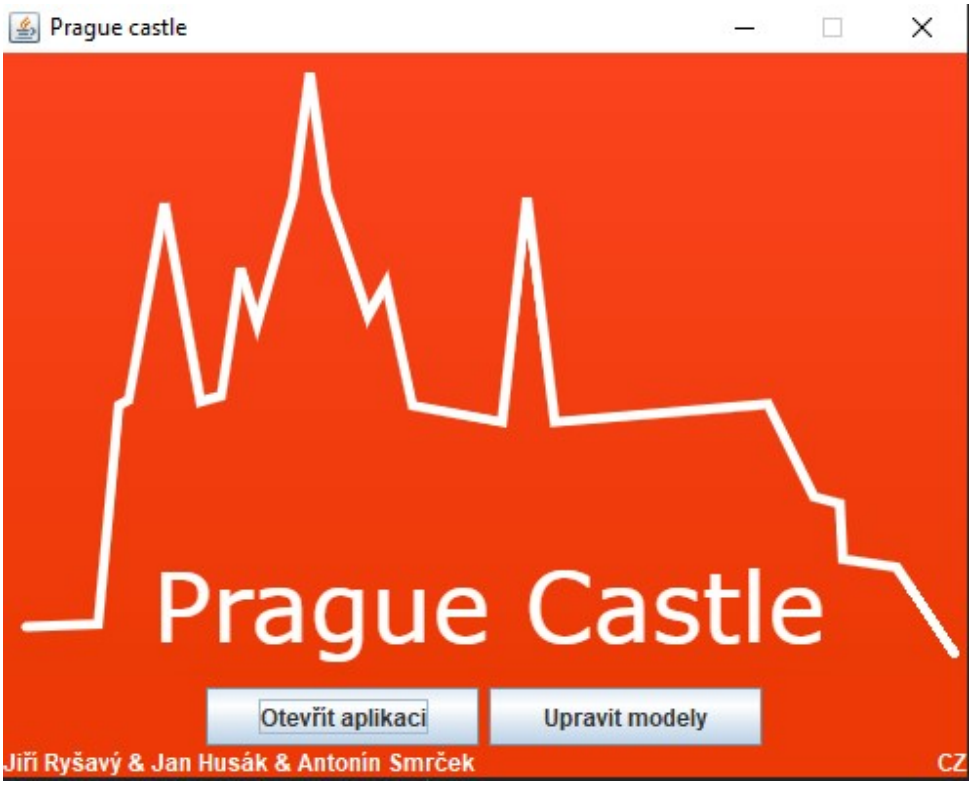
Pro implementaci části view využijeme část Javy `javax.swing.*`. Komponenta Swing dává možnost tvorby uživatelských rozhraní. Základem je okno JFrame, do kterého můžeme přidávat komponenty jako jsou například tlačítka, textová pole či zaškrtávací políčka.
Pro naši aplikaci vytvoříme celkem tři okna. První okno reprezentuje menu. Zde se dá měnit jazyk aplikace, otevřít webová aplikace a spustit editor modelů. Naše implementace je vidět na obrázku 3.3.
Ve druhém okně je samotný editor. Je rozdělený do několika sekcí. V levé části jsou všechny modely, které už v aplikaci jsou. V hlavní části jsou parametry, které lze u jednotlivých objektů měnit. Ve spodní části je pak přepínání režimů editoru, možnost zobrazit úpravu popisků a uložení změn. Aplikace nic nepočítá, dokud uživatel neodešle příkaz pro uložení. Jediné co se mění jsou políčka, která má uživatel přístupná, podle zvoleného režimu. Implementace našeho editoru vypadá takto 3.4
Poslední okno potom slouží pro úpravu informačních panelů objektů. Jsou zde pole pro zadávání jednotlivých textových částí. Pro přidávání odkazů je zde jedno pole, kde se odkazy musí oddělit s pomocí čárek 3.5. V aplikaci také odchytáváme všechny možné výjimky a upozorňujeme na ně uživatele s pomocí chybových hlášení. Nejdůležitější jsou chyby navázání komunikace s FTP serverem, nebo libovolně chybějící soubor.



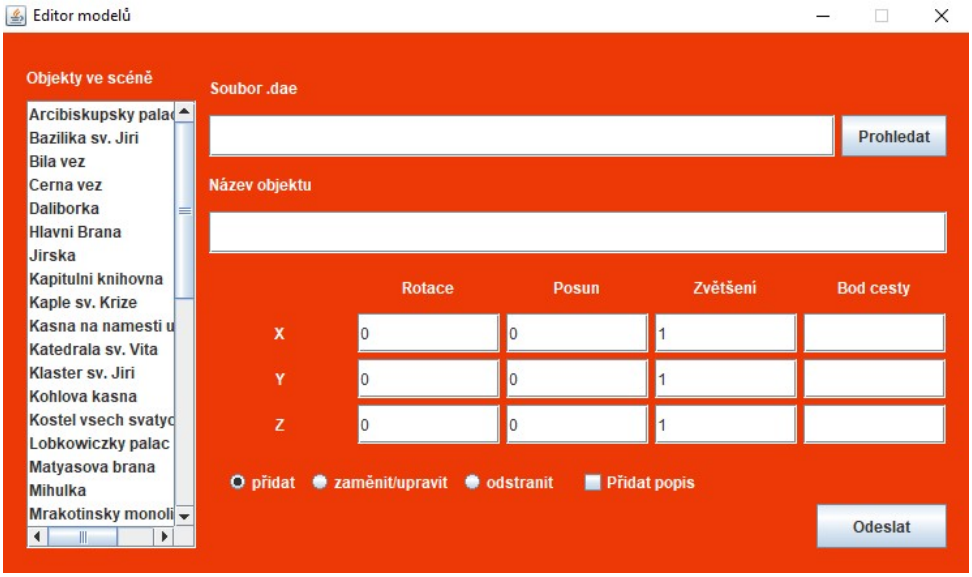
Obrázek 3.1: Příklad provedení přidání objektu.



Obrázek 3.2: Příklad provedení odebrání objektu.



Obrázek 3.3: Vizuální implementace menu.



Obrázek 3.4: Vizuální implementace editoru.

Obrázek 3.5: Vizuální implementace editoru informačního panelu.

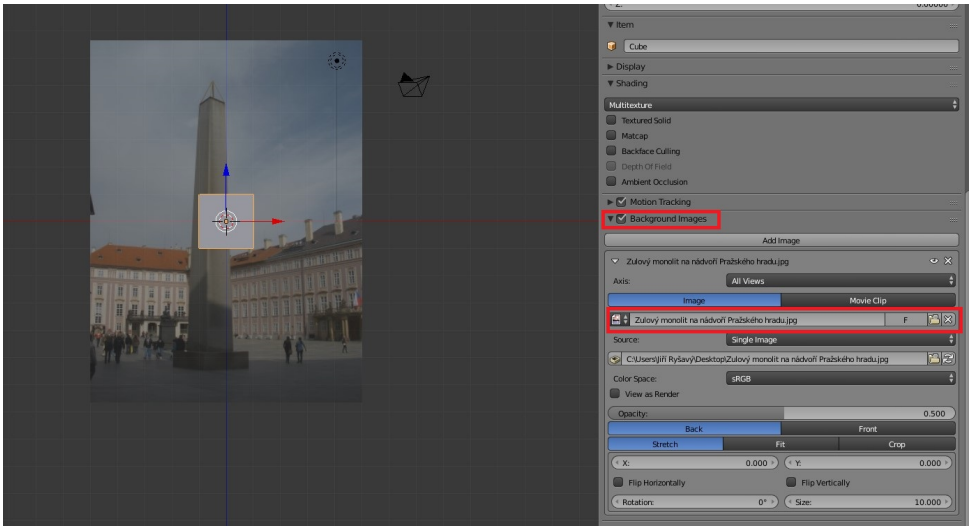
Kapitola 4

3D modelování

Pro 3D modelování využíváme open source program blender[9]. Počátkem tohoto roku vyšla nová verze programu, která rozšířila možnosti. Změnila však i značnou část uživatelského rozhraní. Proto v této práci budeme pracovat stále s verzí 2.79. Tento program nám umožňuje veškerou manipulaci, kterou potřebujeme. Umožňuje nám manipulovat jednotlivými vertexy, upravovat UV souřadnice i tvořit normálové mapy.

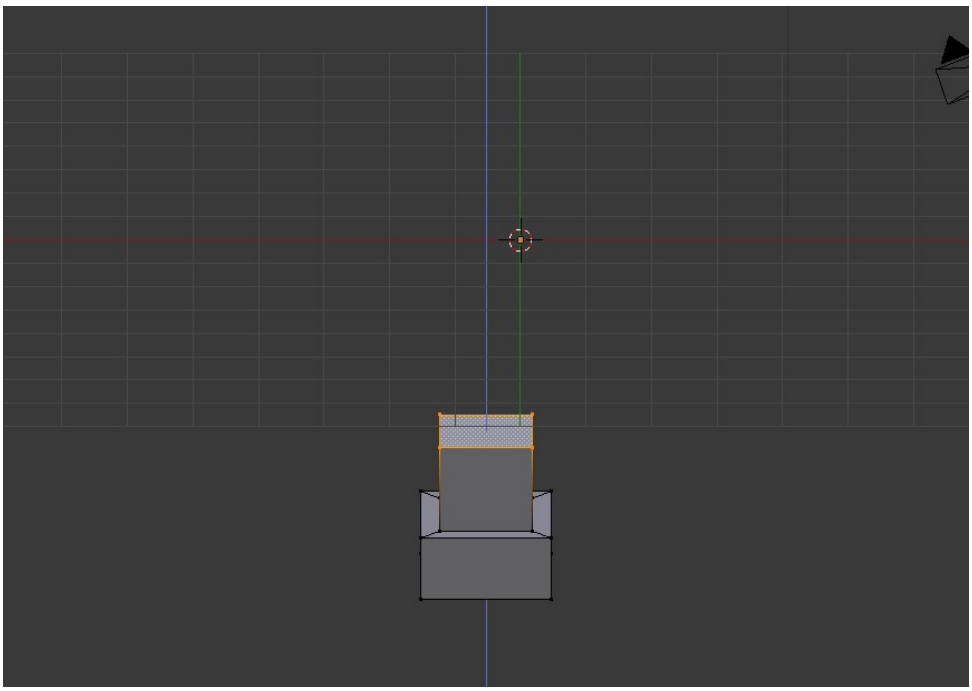
Pokud chceme vytvářet objekty podle předlohy, přidáme si referenční snímky do pozadí. 4.1 Přepneme se do editačního režimu. V tomto režimu máme možnost posunovat jednotlivé vertexy, upravovat je a i přidávat nové. S pomocí manipulátoru upravujeme vertexy do pozic, aby připomínali referenční objekt. Nejčastěji používaným nástrojem bude nástroj `extrude`, který nám umožňuje extrahovat plochu do nového rozměru 4.3. Objekt předlohy se nesnažíme dělat úplně přesně detailně. Pouze se snažíme držet tvar objektu. V případě, kdy je objekt tvořen více složitými částmi, přidáme další objekty dle potřeby. Tyto objekty poté můžeme spojit v hierarchii objektů 4.2.
Jakmile máme objekt, začneme přidávat detaily pro zaoblené hrany. Pro tvoření těchto detailů se používají modifikátory. Modifikátory nám umožňují rozdělovat plošky pro přidání detailu, vytvářet pole pro opakující se objekty, rotovat objekty a spoustu dalších. Tímto způsobem lze do objektů přidat detail, aniž by se snížila vyobrazovací rychlost. Modifikátory totiž nevytváří geometrii navíc, dokud se nepotvrdí. Je možné i řetězit několik modifikátorů na sebe 4.4.
Po domodelování tvaru objektu si označíme švy 4.5. Švy jsou hrany, které nám definují, jak se objekt rozbalí do UV souřadnic pro texturování. Následně objekt rozbalíme 4.6. Pro texturování se využívá pravé okno, kde máme

zobrazené UV souřadnice. Můžeme zde otevřít soubor, který používáme jako texturu.
Pokud nemáme reálnou fotografii materiálu, můžeme využít takzvané kreslení textur, kde s pomocí štětce nanášíme na plátno reprezentující texturu 4.7.
Posledním krokem je vytvoření materiálu objektu. Materiál definuje, jak objekt reaguje na světlo a také celkový vizuál objektu. Materiál obsahuje dvě základní složky. Difuzní odraz a spekulární odraz. S pomocí těchto složek nastavíme, jak materiál reaguje 4.8. Dále se přepneme do záložky pro texturování a objektu přidělíme dříve vytvořenou texturu.
Nyní máme objekt vytvořený a můžeme ho vyexportovat do COLLADA souboru 4.9 4.10.

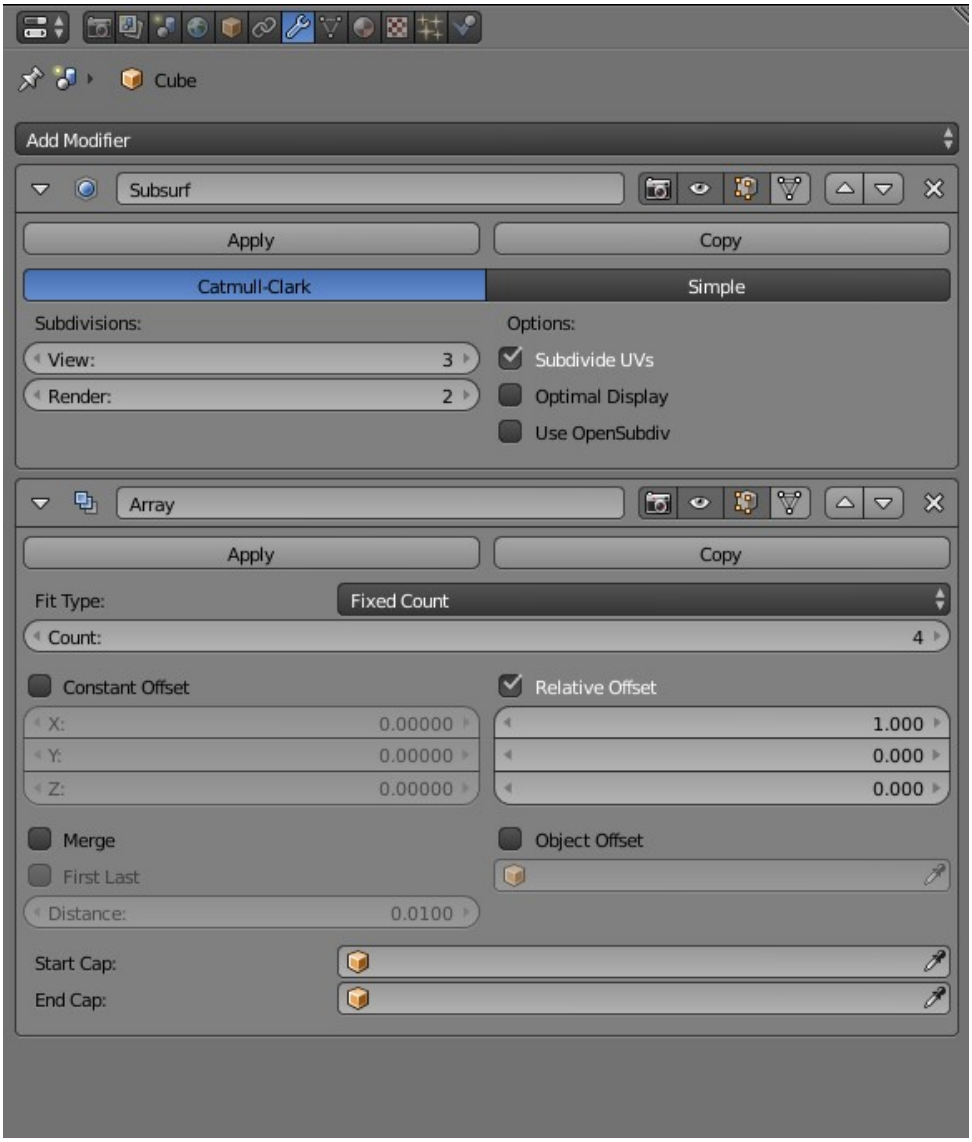


Obrázek 4.1: Nastavení obrázku do pozadí editoru.

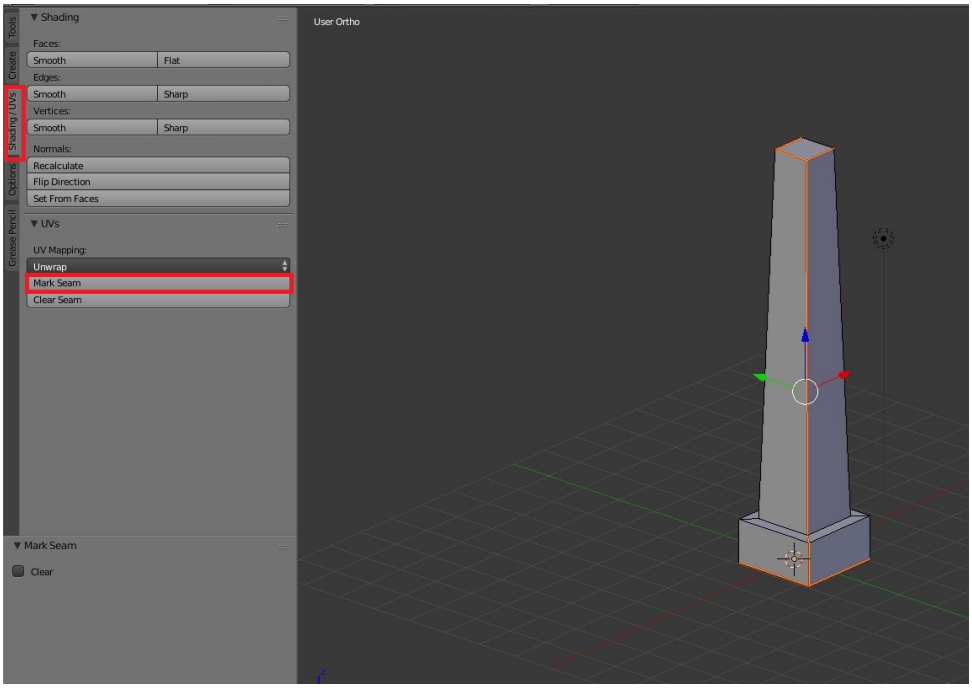
Obrázek 4.2: Více objektů sdružených jako jeden.



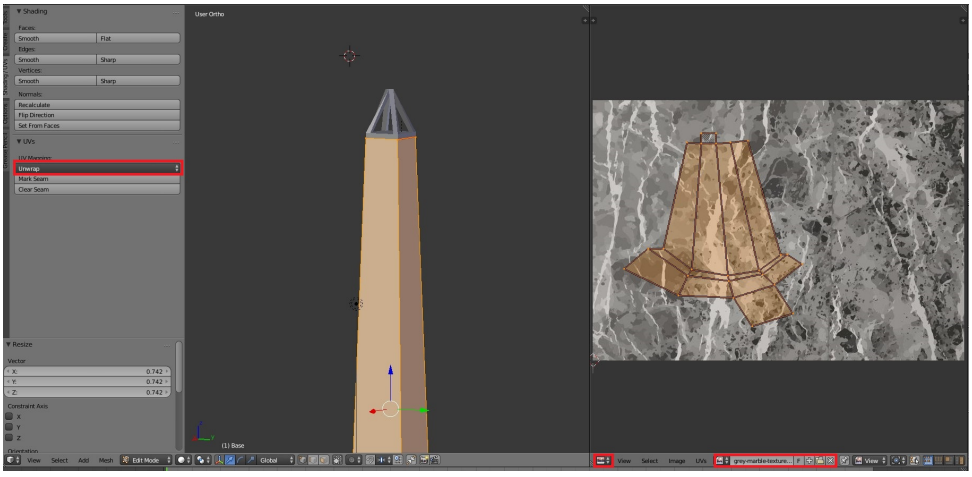
Obrázek 4.3: Použití nástroje extruze.



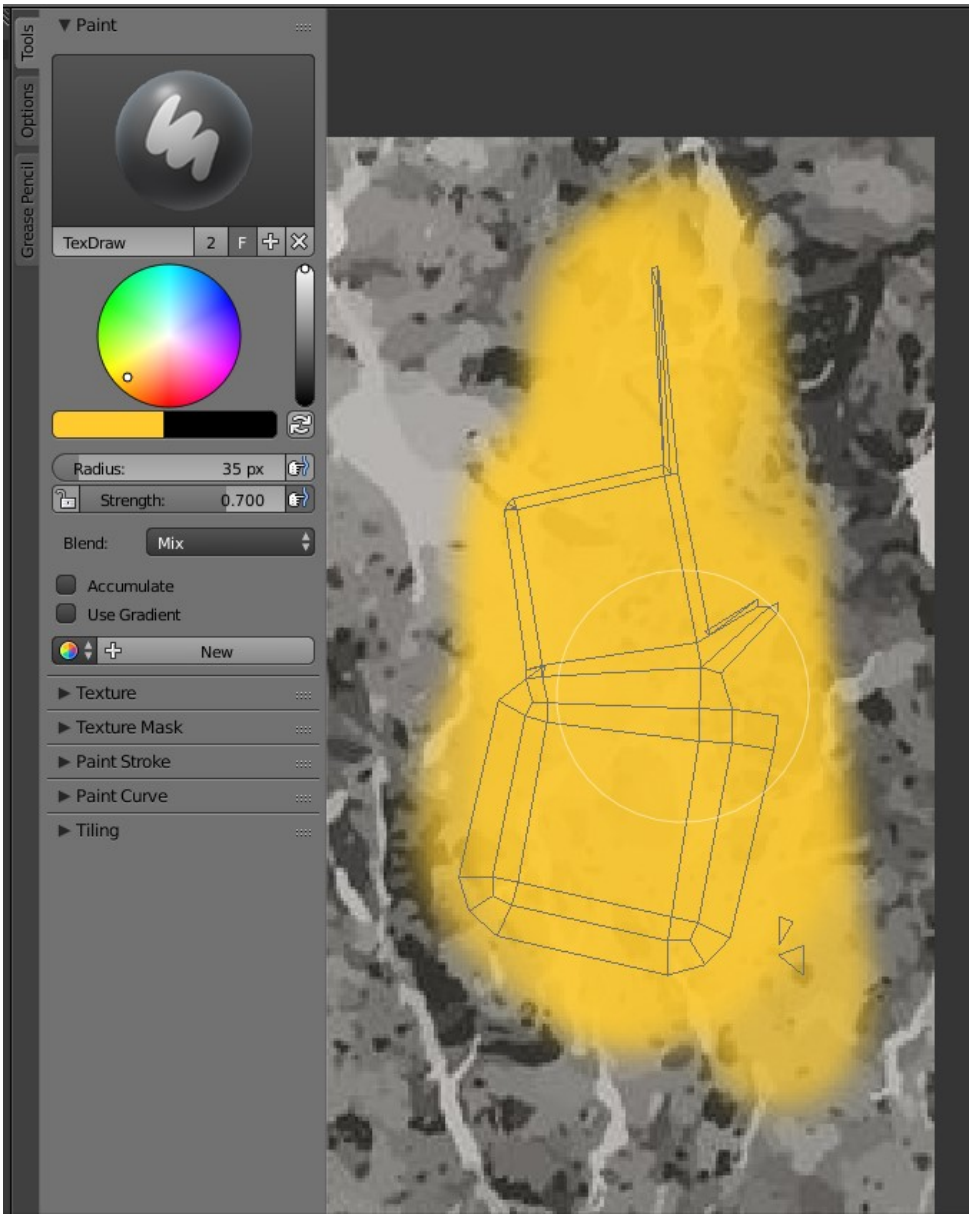
Obrázek 4.4: Záložka modifikátorů.



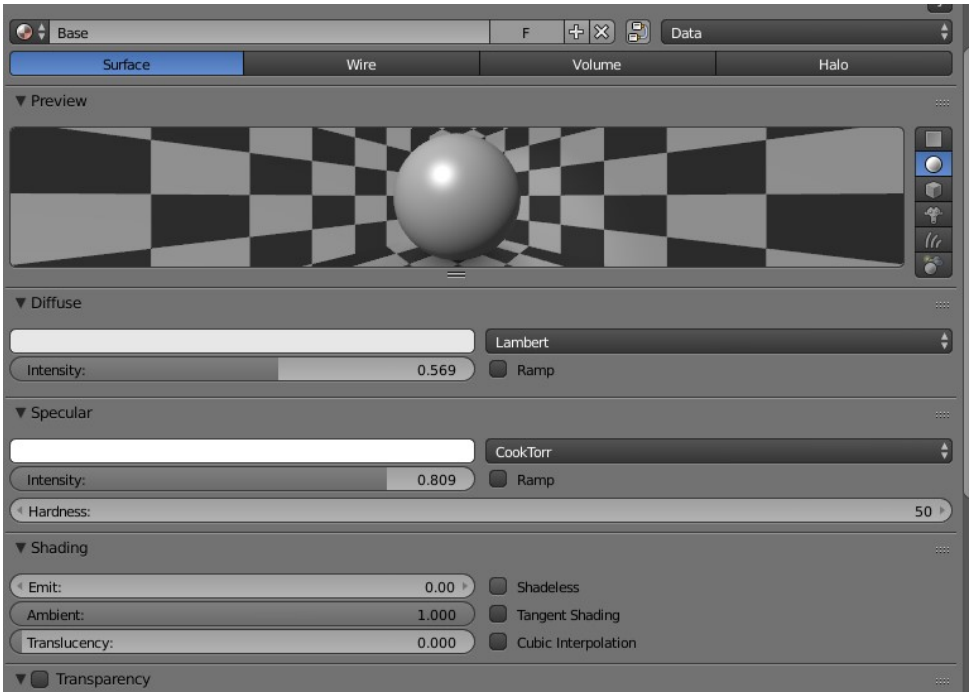
Obrázek 4.5: Označení švů.



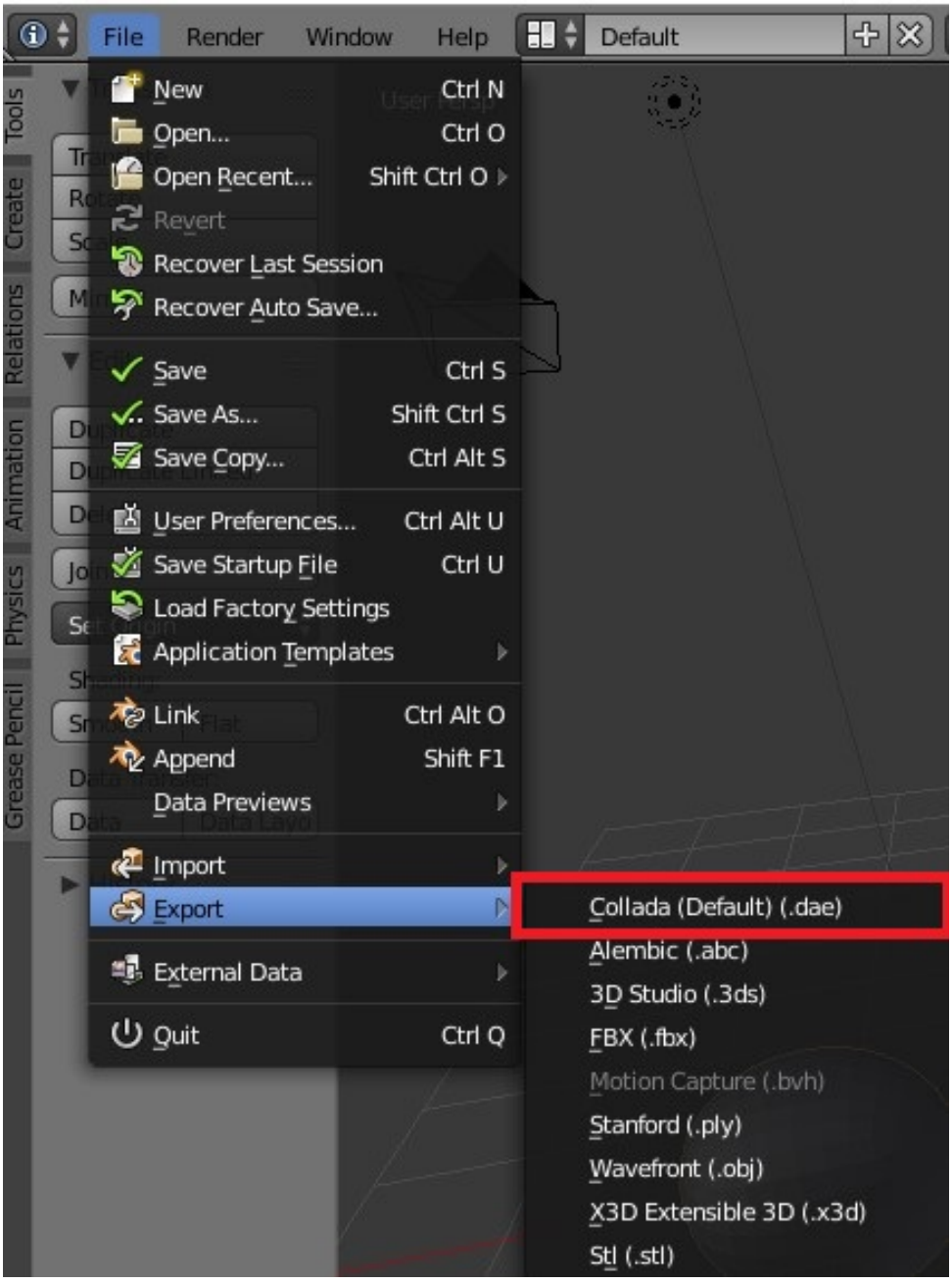
Obrázek 4.6: Texturování.



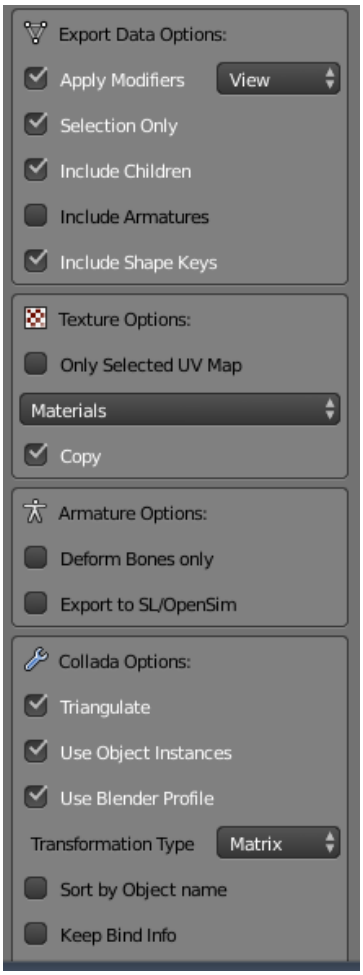
Obrázek 4.7: Malování textury na plátno.



Obrázek 4.8: Záložka pro úpravu materiálu.



Obrázek 4.9: Export.



Obrázek 4.10: Nastavení exportu.

Kapitola 5

Uživatelské testování

Pro uživatelské testování jsme vytvořili scénář, kdy má uživatel před sebou počítač s editorem a jedním vyexportovaným objektem v souboru COLLADA. Uživatel dostal informaci, kde se objekt nachází a měl jej nejprve přidat do scény na jím zvolené místo. Následně měl objekt libovolně upravit a nakonec objekt ze scény smazat. Testování proběhlo u tří uživatelů a jejich reakce byly zaznamenány.

5.1 Změny na základě uživatelského testování

V původním řešení byla úprava informačního panelu ve vyjíždějícím bočním menu. Zde si uživatelé stěžovali na malá políčka pro zobrazení textu. Proto na základě této připomínky je úprava informačního panelu otevírána v novém modálním okně.
Další připomínkou byla nepřehlednost objektů ve scéně. Původně byly zobrazovány podle data přidání. Na základě této žádosti jsme objekty v aplikaci seřadili abecedně. A při přidání nového objektu se nejdříve abecedně zařadí a až poté zapíše do konfiguračního souboru.
Jiná námitka byla vedena na nepřehlednost políček, která jsou aktuálně potřebná. V souvislosti s tímto jsme doimplementovali deaktivaci a aktivaci políček podle zvoleného režimu.
Ostatní námitky se týkaly spíše designového rázu. Týkaly se konkrétně barevné kombinace a velikosti textů. Na tyto připomínky jsme nepřistoupili. Barevná kombinace je volena podle designu původní webové aplikace. Velikosti textů se neupravily z důvodu špatné manipulace s java uživatelskými

rozhraními. V případě, kdy se změní velikost textů, se okno vytvořené přes javax.swing rozsype.

Kapitola 6

Závěr

Cílem práce bylo vytvoření funkčního editoru webové aplikace. Editor měl umožnit přidávání objektů, jejich úpravu a odstraňování. Otestování editoru uživateli a oprava případných nedostatků. Dalším cílem bylo doplnění chybějících částí modelu a sepsání uživatelské příručky.
V teoretické části jsme si popsali, které části z webové aplikace jsou pro naši implementaci důležité. Podrobně jsme rozebrali použité typy souborů a jejich užití v našem editoru.
V implementační části jsme si popsali, jak jednotlivé části aplikace fungují. Uvedli jsme několik příkladů použití knihoven. Dále zde jsou uvedeny návrhy uživatelského rozhraní a snímky z něj.
V další části jsme si ukázali, jak vytvořit objekt do modelu. Také jsme předvedli celkovou práci s 3D modelářem blender.
Dále jsme si popsali, jak proběhlo uživatelské testování a jaký byl daný testovací scénář. Dále jsme sepsali úpravy, které proběhly na základě uživatelského testování a jaké ne.
Nakonec byla sepsána uživatelská příručka, kterou lze nalézt v příloze.

Příloha A

Uživatelská příručka

Uživatelská příručka je určena pro výuku práce s aplikací. Popisuje veškeré funkčnosti, které daná aplikace má a jak s nimi zacházet.

A.1 Příručka

A.1.1 Menu

Menu obsahuje dvě tlačítka a jeden přepínač jazyka. Lavé tlačítko s popiskem "Otevřít aplikaci"("Open application") slouží k otevření webové aplikace. Webová aplikace se otevře v defaultním prohlížeči, nastaveném v operačním systému. Druhé tlačítko spouští editor modelu. V pravém dolním rohu se nachází přepínač jazyka. Po kliknutí na něj se aplikace přepne do druhého jazyka.A.1

A.1.2 Přidávání objektů

V případě, že chceme přidat objekt do scény, potřebujeme mít soubor s modelem, který chceme přidat. Dále potřebujeme znát místo, kam objekt chceme přidat. Soubor můžeme nalézt s pomocí prohlížeče souborového

systému A.2. Dále je třeba zadat název nového objektu a nakonec jednotlivé transformační parametry. Pro orientaci je možné procházet objekty již ve scéně a načítat tak jejich transformační parametry A.3. Jakmile máte vše nastavené dle svých představ, klepněte na tlačítko odeslat. Tím se provede zápis na webový server.

A.1.3 Odebírání objektů

Při odebírání objektů je třeba zvolit mód odebírání objektů ve spodní části. Dále pak v levém panelu zvolíme, který objekt chceme odebrat A.4. Po kliknutí na tlačítko odeslat se ještě program ujistí, že opravdu chcete odebrat daný objekt A.5 a při souhlasu se objekt vymaže.

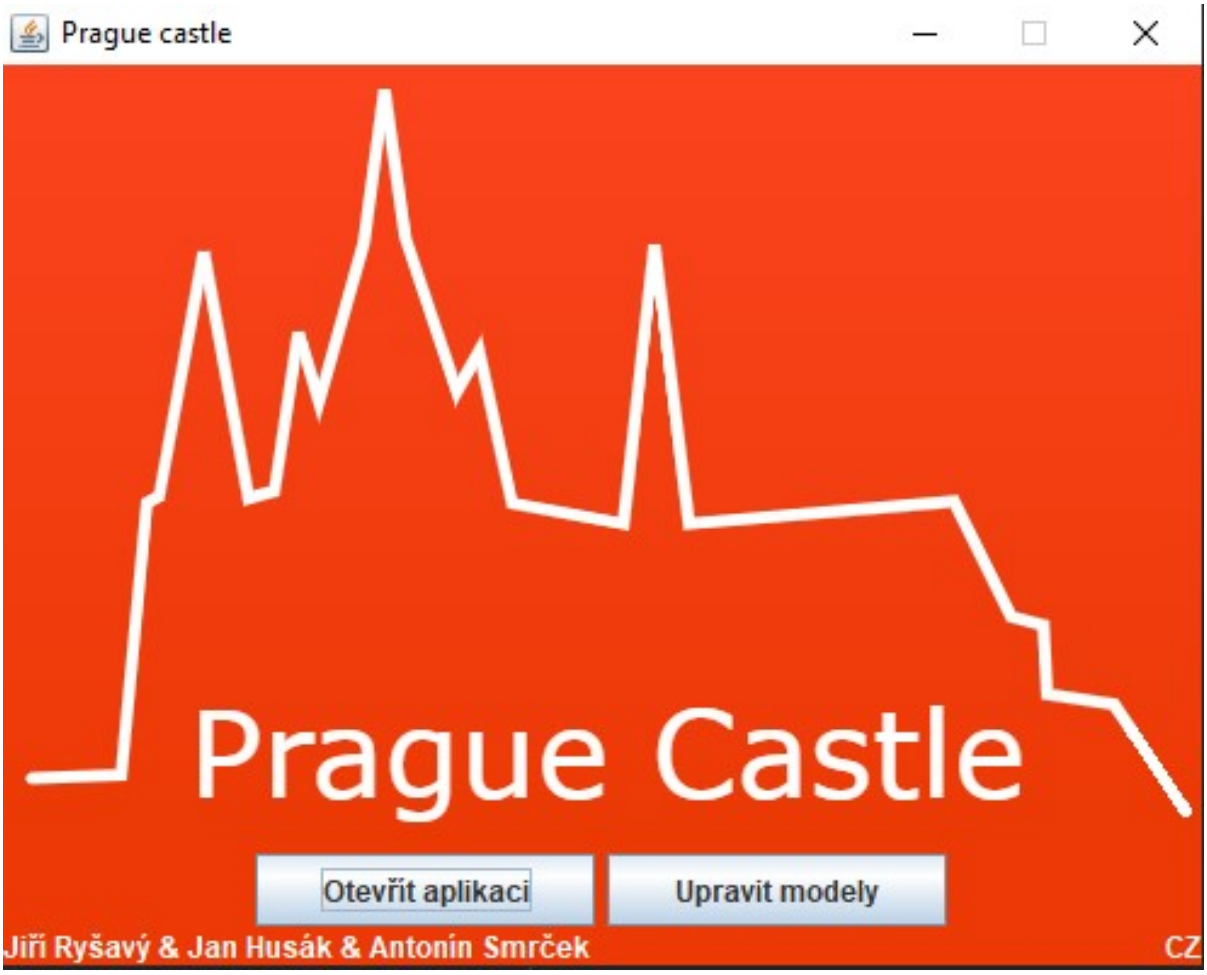
A.1.4 Výměna objektu za objekt

Pokud chceme nahradit objekt, který již ve scéně je, za nový, využijeme mód výměny A.6. Zvolíme mód ve spodní části editoru. Nyní máme přístupné všechny části kromě názvu. Obdobně jako při přidávání zvolíme soubor s modelem. Dále v levém panelu zvolíme objekt, který budeme měnit. Nyní stačí pouze odeslat a daný objekt se vymění.

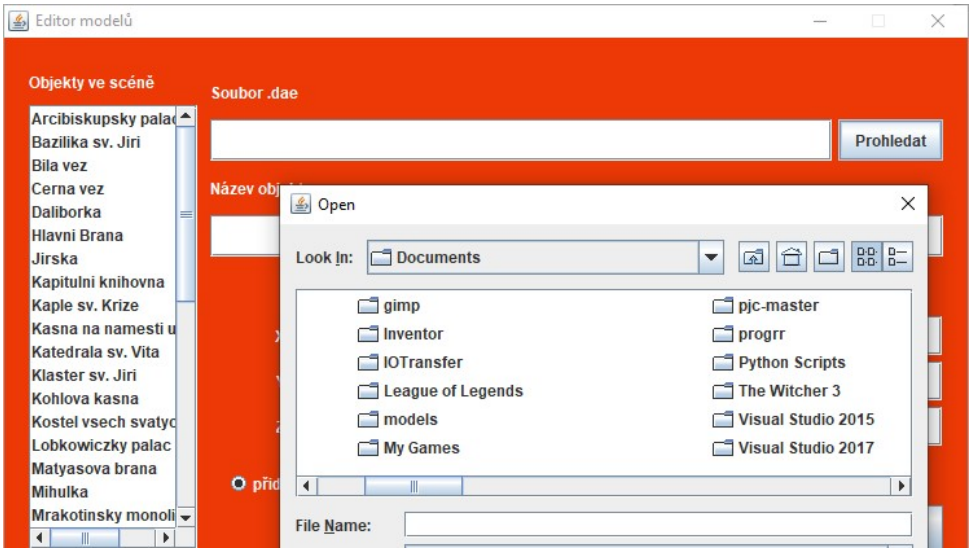
A.1.5 Úprava informačního panelu

Pokud chceme upravovat, nebo přidat informační panel objektu, zaškrtneme políčko "přidat popisek"v editoru A.7. Otevře se editor pro informační panel A.8. V tomto editoru postupně vyplníme informace. Stačí vyplnit pouze ty části, které chcete použít. Odkaz na obrázek slouží k zobrazení obrázku v informačním panelu. Popis k obrázku slouží jako alternativa, pokud webový odkaz obrázku nebude dostupný. Nadpis reprezentuje nadpis informačního panelu. Popisující text je text, který se zobrazí na stránce. Jelikož se jedná o html soubory, je zde možné psát libovolný html text, který se upraví podle možností. Odkazy jsou zdroje k objektu. Může jich být více, je však nutné je oddělit dvojicí čárky a mezery. Všechny části jsou různé pro anglickou a českou verzi popisku. Je tedy možné mít dva různé panely pro jeden objekt. Soubor se vygeneruje po kliknutí na "Odeslat"v editoru objktů.

A.2 Obrázky

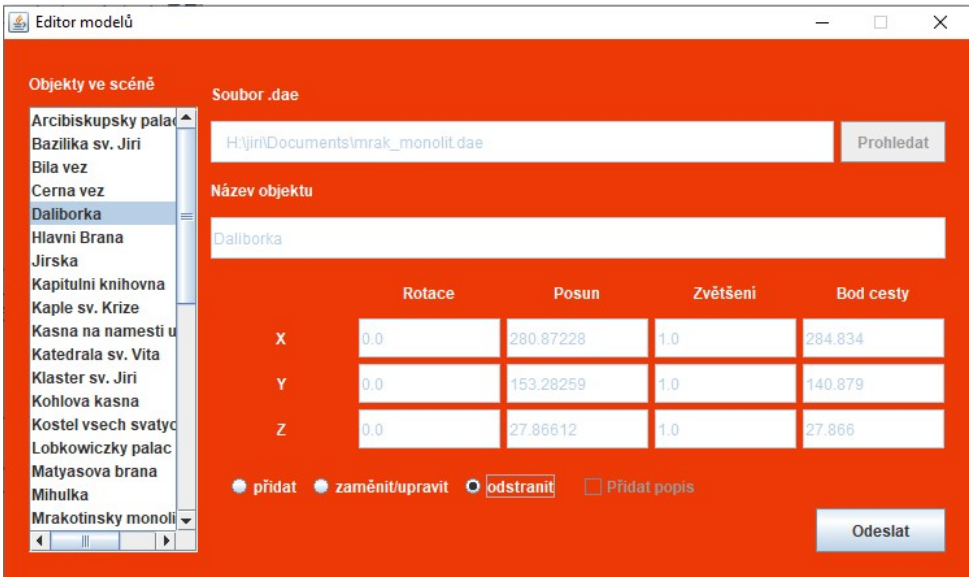


Obrázek A.1: Hlavní nabídka.

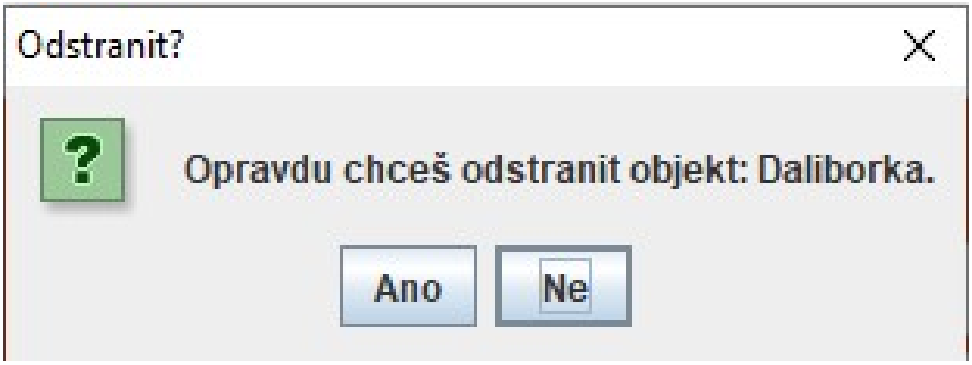


Obrázek A.2: Prohlížeč souborového systému.

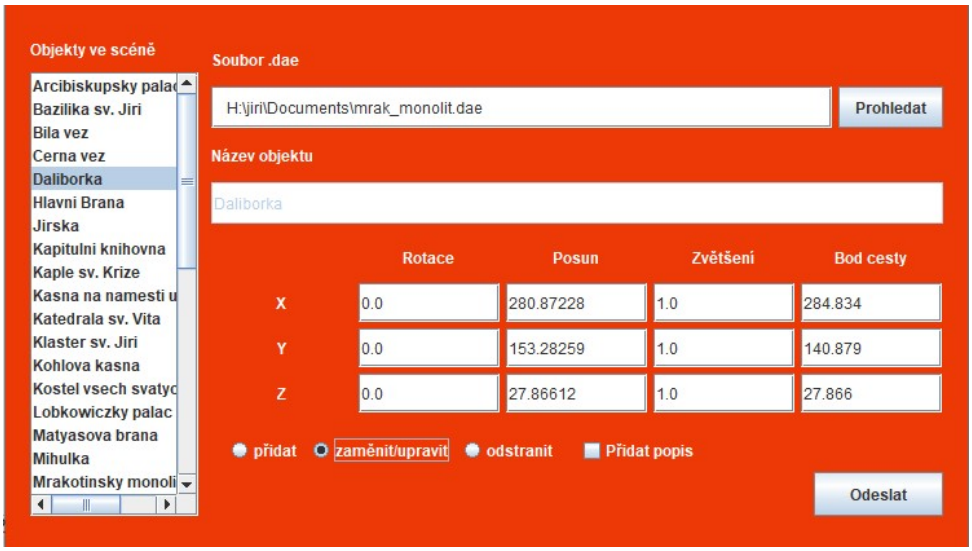
Obrázek A.3: Přepínáním objektů v levém panelu se načítají jejich parametry.



Obrázek A.4: Odstraňování objektu.



Obrázek A.5: Vyskakovací okno pro ujištění odstranění.



Obrázek A.6: Okno v Módu výměny objektů.

Obrázek A.7: Otevření editoru popisků.

Obrázek A.8: Editor popisků/informačních panelů.

Příloha B

Použité zkratky

- COLLADA - COLLAborative Design Activity
- CPU - Control Process Unit
- FTP - File Transfer Protocol
- HTML - Hypertext Markup Language
- HTTP - Hypertext Transfer Protocol
- JAMA - Java Matrix
- JSON - JavaScript Object Notation
- MVC - Model View Controller
- RAM - Random Access Memory
- SMPT - Simple Mail Transfer Protocol
- SNTP - Simple Network Time Protocol
- XML - Extensible Markup Language

Příloha C

Obsah přiloženého média

- Thesis.pdf – tato práce v PDF souboru
- editor.jar - Spustitelný soubor editoru připojený na localhost FTP server s uživatelským jménem admin a heslem admin
- editor.zip - zdrojové kódy editoru
- server.zip - zdrojové kódy serveru s novým modelem

Příloha D

Zadání práce

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

I. OSOBNÍ A STUDIJNÍ ÚDAJE

Příjmení: **Ryšavý** Jméno: **Jiří** Osobní číslo: **457973**

Fakulta/ústav: **Fakulta elektrotechnická**

Zadávající katedra/ústav: **Katedra počítačové grafiky a interakce**

Studijní program: **Otevřená informatika**

Studijní obor: **Počítačové hry a grafika**

II. ÚDAJE K BAKALÁŘSKÉ PRÁCI

Název bakalářské práce:

Kompletace 3D modelu Pražského hradu

Název bakalářské práce anglicky:

Completion of Prague Castle 3D model

Pokyny pro vypracování:

Seznamte se s existující aplikací - https://dp.jan-husak.cz/ - pro vizualizaci Pražského hradu (autor J. Husák) a doplňte ji:
- Vytvořte samostatný program, s jehož pomocí bude možné automaticky přidávat konkrétní nové části Pražského hradu do stávajícího modelu. Program bude schopen buď přidávat zcela nové objekty nebo nahradit existující.
- Pomocí svého programu doplňte do 3D modelu budovy a objekty tak, aby výsledkem byl kompletní Pražský hrad, včetně potřebných stanovišť, zajímavých lokací a příslušných informačních textů.
- Pokud budou některé druhy aktualizací vyžadovat ruční zásahy do dat systému, vytvořte pro uživatele detailní návod (kuchařku), jak potřebných změn docílit.
- Svůj program včas podrobte uživatelskému testování a podle výsledků testování zrealizujte další zlepšení.

Seznam doporučené literatury:

1) Zara Jiri: Web-Based Historical City Walks: Advances and Bottlenecks. PRESENCE: Teleoperators and Virtual Environments. 2006, vol. 15, no. 3, p. 262-277. ISSN 1054-7460.
2) Smrček Antonín: Vizualizace Pražského hradu. Diplomová práce, FEL ČVUT. 2016.
3) Husák Jan: Interaktivní prezentace Pražského hradu. Diplomová práce (nedokončeno), FEL ČVUT. 2017.

Jméno a pracoviště vedoucí(ho) bakalářské práce:

prof. Ing. Jiří Žára, CSc., katedra počítačové grafiky a interakce FEL

Jméno a pracoviště druhé(ho) vedoucí(ho) nebo konzultanta(ky) bakalářské práce:

Datum zadání bakalářské práce: **30.10.2018** Termín odevzdání bakalářské práce: _____

Platnost zadání bakalářské práce: **20.09.2020**

prof. Ing. Jiří Žára, CSc. podpis vedoucí(ho) práce	podpis vedoucí(ho) ústavu/katedry	prof. Ing. Pavel Ripka, CSc. podpis děkana(ky)

III. PŘEVZETÍ ZADÁNÍ

Student bere na vědomí, že je povinen vypracovat bakalářskou práci samostatně, bez cizí pomoci, s výjimkou poskytnutých konzultací. Seznam použité literatury, jiných pramenů a jmen konzultantů je třeba uvést v bakalářské práci.

8.5.2020
Datum převzetí zadání

Podpis studenta

Příloha E

Literatura

[1] Antonín Smrček. Vizualizace Pražského hradu. 2016. Diplomová práce. ČVUT v Praze, Fakulta elektrotechnická, katedra počítačové grafiky a interakce.

[2] Jan Husák. Interaktivní prezentace Pražského hradu. 2017. Diplomová práce. ČVUT v Praze, Fakulta elektrotechnická, katedra počítačové grafiky a interakce.

[3] Introducing JSON[online]. JSON. [vid. 20. 4. 2020]. `https://www.json.org/json-en.html`

[4] Sony Computer Entertainment[online]. COLLADA – Digital Asset Schema Release 1.4.1. Dostupné z: `https://www.khronos.org/files/collada_spec_1_4.pdf`

[5] Eric Shepherd[online]. HTML: Hypertext Markup Language. In: MDN web docs[online]. Mozilla and individual contributors. 2013.[vid. 20. 4. 2020] Dostupné z: `https://developer.mozilla.org/en-US/docs/Web/HTML`

[6] MathWorks, NIST. JAMA: A Java Matrix Package[software]. August 1998[přístup: 11. 9. 2000]. Dostupné z: `https://math.nist.gov/javanumerics/jama/`. [Požadavky na systém: Windows 8/7/Vista/XP/2000, Windows Server 2008/2003. Intel Pentium 166 MHz, 64 RAM, 98 MB místa na disku.]

[7] Apache Commons Net. Commons NET[software]. 1998[přístup: 28. 2. 2008]. Dostupné z:

http://commons.apache.org/index.html. [Požadavky na systém: Windows 8/7/Vista/XP/2000, Windows Server 2008/2003. Intel Pentium 166 MHz, 64 RAM, 98 MB místa na disku.]

[8] Yidong Fang. json-simple[software]. November 2008[přístup: 10. 1. 2009]. Dostupné z:
https://github.com/fangyidong/json-simple. [Požadavky na systém: Windows 8/7/Vista/XP/2000, Windows Server 2008/2003. Intel Pentium 166 MHz, 64 RAM, 98 MB místa na disku.]

[9] Blender foundation. Blender[software]. January 1994[přístup: 13. 10. 2002]. Dostupné z:
https://www.blender.org. [Požadavky na systém: 64-bit dual core 2Ghz CPU s podporou SSE2, 4GB RAM, 1280768 monitor, Myš, Grafická karta s 1GB RAM, OpenGL 3.3]