**Šablona závěrečné práce  
studenta Unicorn Vysoká škola**

*Tato první stránka šablony není součástí bakalářské práce.*

*Tato šablona slouží jako vzorová šablona závěrečných prací studentům Unicorn Vysoké školy. Závěrečná práce musí obsahovat všechny náležitosti uvedené v této šabloně.*

*Nesplnění této podmínky může být považováno za důvod pro nepřipuštění závěrečné práce k obhajobě (nebo případně k vrácení práce od obhajoby k přepracování).*

*Další informace a pokyny k vypracování závěrečná práce naleznete na webových stránkách. Vše potřebné se také dozvíte v rámci předmětu Bakalářský seminář.*

*Při zpracování této šablony bylo použito písmo Cambria, 11pt. pro text a písmo Calibri pro nadpisy.*

**UNICORN VYSOKÁ ŠKOLA S.R.O.**

BAKALÁŘSKÁ PRÁCE

**2023 Valerij Šlovikov**

**UNICORN VYSOKÁ ŠKOLA S.R.O.**

**Softwarový vývoj**

****

**BAKALÁŘSKÁ PRÁCE**

**Název práce (přesně podle zadání)**

**Autor BP:** Valerij Šlovikov

**Vedoucí BP:** Ing. Michal Gregor

*Vzor:* ***ZADÁNÍ ZÁVĚREČNÉ PRÁCE*** *– originál, kopie originálu, naskenovaná podoba – dle jednotlivých forem (originál, 2 x kopie, elektronická verze)*

*Vzor:* ***ČESTNÉ PROHLÁŠENÍ*** *– prohlášení o samostatném vypracování závěrečné práce, datum a vlastnoruční podpis (v každém výtisku práce)*

**Čestné prohlášení**

Prohlašuji, že jsem svou bakalářskou práci na téma ....................... vypracoval/a samostatně pod vedením vedoucího bakalářské práce a s použitím výhradně odborné literatury a dalších informačních zdrojů, které jsou v práci všechny citovány a jsou také uvedeny v seznamu použitých zdrojů.

Jako autor/ka této bakalářské práce dále prohlašuji, že v souvislosti s jejím vytvořením jsem neporušil/a autorská práva třetích osob a jsem si plně vědom/a následků porušení ustanovení § 11 a následujících autorského zákona č. 121/2000 Sb.

Dále prohlašuji, že odevzdaná tištěná verze bakalářské práce je shodná s verzí, která byla odevzdána elektronicky.

V……………………. dne ……….. …….……………………………

(Jan Novák)

*Vzor:* ***PODĚKOVÁNÍ*** *vedoucímu BP, konzultantům, odborníků, spolupracovníkům za poskytnuté rady a podkladové materiály apod.) –* ***není povinné***

**Poděkování**

Např: Děkuji vedoucímu bakalářské práce Jméno Příjmení (i s tituly) za účinnou metodickou, pedagogickou a odbornou pomoc a další cenné rady při zpracování mé bakalářské práce…

*Vzor:* ***PRVNÍ ČÍSLOVANÁ STRANA*** *– číslice na první číslované straně se určí podle počtu předchozích stran, počínaje Titulní stranou, tzn. že pokud jsou řazené všechny dané strany – Titulní strana, Zadání (2 strany), Čestné prohlášení a Poděkování – je první číslovaná strana stranou 6.*

**

**Název práce v českém/slovenském jazyce Název práce v anglickém jazyce**

**Abstrakt**

Abstrakt česky. Abstrakt krátce a výstižně charakterizuje obsah závěrečné práce. Zpravidla obsahuje informace o stanovených cílech, použitých metodách, postupu řešení a výsledcích výzkumu. Může obsahovat krátkou informaci o použitých zdrojích. Délka abstraktu je zpravidla 100–500 slov.

Klíčová slova: klíčová slova práce, minimálně 5, maximálně 10

# Abstrakt

Tato bakalářská práce se zabývá možnostmi a využitím 3D grafiky na webu, především s ohledem na technologie WebGL, WebGL2 a WebGPU. Cílem je porovnat jejich funkční rozdíly, zhodnotit podporu různými prohlížeči, bezpečnostní aspekty a demonstrovat reálné aplikace včetně ukázkových projektů. Součástí práce je také průzkum knihoven, jako jsou Three.js a Babylon.js, se zaměřením na jednoduchost a efektivitu práce. Zvláštní pozornost je věnována parallax mappingu a jeho využití ve spojení se šroubovicí, jakož i potenciálu WebGPU pro akceleraci AI výpočtů přímo v prohlížeči.

Praktická část se zaměřuje na tvorbu několika ukázkových aplikací, včetně vykreslení jednoduchých tvarů, „Hello World“ příkladů a pokročilejších scén využívajících parallax mapping. Dále je popsána implementace canvas komponenty v rámci frameworku uu5 a provádí se analýza výkonu jednotlivých řešení. Výstupem je soubor poznatků a doporučení, jak efektivně navrhovat a vyvíjet 3D aplikace na webu.

*Klíčová slova:* 3D grafika, WebGL, WebGL2, WebGPU, parallax mapping, Three.js, Babylon.js, AI, výkon, webové prohlížeče

This bachelor’s thesis focuses on the possibilities and usage of 3D graphics on the web, primarily examining WebGL, WebGL2, and WebGPU. The goal is to compare their functional differences, evaluate browser support, explore security considerations, and demonstrate real-world applications via sample projects. The research also explores various libraries such as Three.js and Babylon.js, focusing on ease of use and development efficiency. Special attention is given to parallax mapping, its integration with a helix model, and the potential of WebGPU for accelerating AI computations directly in the browser.

The practical part centers around creating several sample applications, including rendering simple shapes, “Hello World” demos, and more advanced scenes featuring parallax mapping. Additionally, the implementation of a canvas component within the uu5 framework is described, accompanied by a performance analysis of the different approaches. The outcome is a set of insights and recommendations for effectively designing and developing 3D applications on the web.

Keywords: 3D graphics, WebGL, WebGL2, WebGPU, parallax mapping, Three.js, Babylon.js, AI, performance, web browsers

**Abstract**

Zde umístěte překlad abstraktu do anglického jazyka. Česky a anglicky psané abstrakty musí být totožné. Student/ka zodpovídá za jazykovou správnost anglického překladu. V případě, že se anglická a česká verze nevejdou na jednu stránku, umístěte celý překlad na samostatnou stránku.

Keywords: klíčová slova v anglickém jazyc

**Obsah**

[Úvod do 3D Grafiky na Webu 10](#_Toc143686539)

[1.1 Popis webGL, webGL2, a webGPU 11](#_Toc143686540)

[1.2 Historie a vývoj 12](#_Toc143686541)

[1.3 Aplikace a význam v moderním webovém designu 13](#_Toc143686542)

[2 Teoretická část (není název kapitoly) 14](#_Toc143686543)

[2.1 Porovnání WebGL, WebGL2 a WebGPU 14](#_Toc143686544)

[2.1.1 Funkční rozdíly 15](#_Toc143686545)

[2.1.2 Podpora prohlížečů 16](#_Toc143686546)

[2.1.3 Bezpečnostní aspekty 17](#_Toc143686547)

[2.2 Knihovny pro Práci s 3D Grafikou 18](#_Toc143686548)

[2.2.1 Three.js, Babylon.js, atd. 18](#_Toc143686549)

[2.2.2 Porovnání jednoduchosti práce s těmito knihovnami 18](#_Toc143686550)

[2.3 Parallax Mapping a Šroubovice 18](#_Toc143686551)

[2.3.1 Co to je a jak funguje 18](#_Toc143686552)

[2.3.2 Využití v 3D grafice 18](#_Toc143686553)

[2.4 Akcelerace AI pomocí WebGPU 18](#_Toc143686554)

[2.4.1 Základy strojového učení na grafických procesorech 18](#_Toc143686555)

[2.4.2 WebGPU a jeho výhody pro AI 18](#_Toc143686556)

[3 Praktická část/Empirická část/Vlastní práce (není název kapitoly) 19](#_Toc143686557)

[3.1 Nadpis úrovně 2 19](#_Toc143686558)

[3.2 Vykreslení Trojúhelníku s Přechodem RGB 21](#_Toc143686559)

[3.2.1 Vytvoření ukázkových projektů v Three.js, WebGL2 a WebGPU 21](#_Toc143686560)

[3.2.2 Srovnání výkonu (FPS) a složitosti kódu 21](#_Toc143686561)

[3.3 Vytvoření Hello World v Různých Technologiích 21](#_Toc143686562)

[3.3.1 Stejně jako výše, ale s jednoduchým "Hello World" příkladem 21](#_Toc143686563)

[3.4 Komponenta v Canvas v uu5 21](#_Toc143686564)

[3.4.1 Vývoj komponenty v uu5 21](#_Toc143686565)

[3.4.2 Vypsání statistik, výkonu 21](#_Toc143686566)

[3.5 Šroubovice s Paralax Mappingem 21](#_Toc143686567)

[3.5.1 Implementace šroubovice s paralax mappingem 21](#_Toc143686568)

[3.5.2 Analýza a srovnání výkonu 21](#_Toc143686569)

[3.6 Experimenty s AI a Grafikou 21](#_Toc143686570)

[3.6.1 (Nad rámec) Experimenty s přímým přístupem ke grafice pro strojové učení 21](#_Toc143686571)

[Závěr 22](#_Toc143686572)

[Seznam použitých zdrojů 23](#_Toc143686573)

[Seznam obrázků (existují-li) 24](#_Toc143686574)

[Seznam grafů (existují-li) 26](#_Toc143686575)

[Seznam příloh (existují-li) 27](#_Toc143686576)

[Příloha A – Název přílohy 28](#_Toc143686577)

[Příloha B – Název přílohy 29](#_Toc143686578)

# Úvod do 3D Grafiky na Webu

3D grafika na webu otevřela nové možnosti pro vývojáře, designéry i koncové uživatele, poskytující bohaté a interaktivní zážitky přímo v internetovém prohlížeči. Již nemusíme čekat na vzdálenou budoucnost, kdy 3D modely, animace a dokonce i celé virtuální světy budou dostupné na dosah kliknutí. Tyto technologie už jsou tady a postupně se stávají standardní součástí moderních webových aplikací.

### 1.1 Proč je 3D grafika na webu důležitá

V době, kdy uživatelé očekávají stále atraktivnější a dynamičtější obsah, představuje 3D grafika výrazné odlišení od běžných 2D webů. Ať už se jedná o prezentační web pro architektonický projekt, online konfigurátor produktu, nebo interaktivní vzdělávací nástroj, 3D modely a animace vtahují uživatele mnohem hlouběji do obsahu. Tím se zvyšuje míra zapojení i délka stráveného času na webu, což může výrazně přispět k úspěchu projektu.

Dříve byla 3D grafika doménou specializovaného softwaru a hardwaru. Uživatelé potřebovali výkonné stroje a programy, jako je Blender, 3ds Max, Maya, případně herní enginy typu Unity či Unreal Engine. Nasazení 3D prvků na webu se často řešilo přes pluginy jako Flash nebo Unity Web Player, což bylo komplikované z pohledu bezpečnosti i uživatelského komfortu. Vstupem technologií jako WebGL, WebGL2 a WebGPU do mainstreamových prohlížečů se 3D grafika stává daleko přístupnější a demokratizovanější, a to jak pro vývojáře, tak pro uživatele.

### 1.2 Popis WebGL, WebGL2 a WebGPU

#### 1.2.1 WebGL

**WebGL (Web Graphics Library)** je API vycházející z OpenGL ES 2.0, které umožňuje vykreslování 2D i 3D grafiky přímo v prohlížeči bez nutnosti instalovat jakékoli externí pluginy. Je výsledkem snah skupiny Khronos Group a poprvé bylo oficiálně vydáno v roce 2011. Od té doby si WebGL našlo cestu do všech hlavních prohlížečů včetně Google Chrome, Mozilla Firefox, Safari a Microsoft Edge.

* **Fungování:**  
  WebGL využívá kódy zvané shadery (vertex a fragment shadery), které běží na grafické kartě (GPU). Programátor píše tyto shadery v jazyce GLSL (OpenGL Shading Language) a skrze JavaScript s nimi komunikuje. Na nízkoúrovňové úrovni tak dochází ke komunikaci mezi prohlížečem a grafickým hardwarem uživatele.
* **Hlavní rysy:**
  + **Vykreslování bez pluginů** – stačí aktuální prohlížeč.
  + **OpenGL ES 2.0** – WebGL přináší na web klíčové funkce známé z mobilní verze OpenGL.
  + **Podpora GPU** – umožňuje akcelerované vykreslování i složitých scén, her a interaktivních projektů.
  + **Omezení** – některé pokročilejší funkce moderních GPU (jako 3D textury nebo některé formy pokročilých shaderů) nebyly v původním WebGL dostupné.
* **Využití v praxi:**
  + Interaktivní grafy, vzdělávací simulace (např. 3D anatomie, fyzikální modely).
  + Základní 3D hry a vizualizace CAD modelů přímo v prohlížeči.
  + Výchozí technologie pro různé knihovny typu **Three.js**, **Babylon.js**, které abstrahují nízkoúrovňovou práci s buffery a shadery.

#### 1.2.2 WebGL2

S vydáním **WebGL2** se podpora 3D grafiky na webu dostala o krok dál. WebGL2 vychází z **OpenGL ES 3.0**, čímž rozšiřuje původní WebGL o celou řadu funkcí, které byly dříve dostupné pouze na nativních platformách.

* **Co nového přináší:**
  + **Pokročilé vykreslovací techniky**: Například transform feedback, který umožňuje zachycovat výsledky z vertex shaderu a dále je zpracovávat.
  + **Instanced rendering**: Možnost vykreslovat opakované instance stejného objektu (např. tisíce stromů v lese) efektivněji.
  + **3D textury a rozšířená správa bufferů**: Poskytuje více možností, jak textury ukládat a pracovat s nimi.
  + **Vylepšené shadery**: Podpora rozšířeného GLSL, které umožňuje použít komplexnější výpočty a funkce přímo na GPU.
* **Kompatibilita:**
  + V současnosti mají hlavní prohlížeče (Chrome, Firefox, Edge) již poměrně stabilní podporu WebGL2, Safari stále postupně dohání.
  + Na mobilních zařízeních může být situace o něco komplikovanější, některá starší zařízení neimplementují OpenGL ES 3.0 dostatečně stabilně.
* **Přínosy v praxi:**
  + **Lepší výkon** u složitých 3D scén (např. masivní scény s mnoha objekty).
  + **Realistické efekty** jako volumetrické světlo, pokročilé stínování či post-processing.
  + **Efektivnější práce s daty** díky vylepšené správě bufferů a paměti.

#### 1.2.3 WebGPU

Nejnovějším přírůstkem v oblasti webové 3D grafiky a GPU akcelerace je **WebGPU**. Stojí za ním opět konsorcium Khronos Group ve spolupráci s dalšími klíčovými hráči. WebGPU se inspiruje moderními grafickými API, jako jsou **Vulkan**, **Metal** (Apple) a **Direct3D 12** (Microsoft), a přináší na web nižší úroveň přístupu k GPU a další výhody, které WebGL/WebGL2 neposkytují.

* **Klíčové vlastnosti:**
  + **Větší kontrola nad hardwarem**: WebGPU je nízkoúrovňové API, které dává vývojářům jemnější řízení nad tím, jak se grafika (a výpočty) na GPU provádí.
  + **Vyšší výkon**: Díky nízkoúrovňové správě paměti, front vykreslování (rendering queues) a asynchronním výpočtům lze dosáhnout lepšího využití GPU.
  + **Podpora výpočtů (compute shaders)**: Kromě grafiky lze WebGPU využívat i pro obecné paralelní výpočty, což je ideální pro úlohy strojového učení (ML) nebo vědeckých výpočtů.
  + **Flexibilita**: Architektonicky vychází z designu „moderních“ API (Vulkan, Metal, Direct3D 12), proto je nastavení pipeline a resource managementu pro zkušené GPU programátory výrazně flexibilnější než WebGL.
* **Stav podpory v prohlížečích:**
  + WebGPU je stále relativní novinka. V Chrome a ve Firefoxu existuje experimentální podpora (často schovaná za příznaky „flags“).
  + Safari a Edge také postupně zavádějí podporu, ale musí se počítat s tím, že není tak stabilní jako u WebGL2.
  + Vývojový ekosystém okolo WebGPU se teprve rodí, nicméně existují už knihovny (např. wgpu, Dawn od Googlu, WebGPU adaptace v Babylon.js) usnadňující práci.
* **Příklady využití:**
  + **Kombinace 3D a AI**: Například rendering scény a zároveň strojové učení pro herní logiku v reálném čase.
  + **Pokročilé grafické efekty**: Realistické stíny, GI (global illumination), fluidní simulace, hair & fur simulace atd.
  + **Výpočetně náročné úlohy**: WebGPU se neomezuje jen na grafiku – lze ho použít pro parallel computing, zpracování velkých datasetů přímo na GPU, generování terénu apod.

#### 1.2.4 Porovnání WebGL/WebGL2 a WebGPU

| **Vlastnost** | **WebGL** | **WebGL2** | **WebGPU** |
| --- | --- | --- | --- |
| **Založeno na** | OpenGL ES 2.0 | OpenGL ES 3.0 | Vulkan, Metal, Direct3D 12 |
| **Pokročilé funkce** | Omezené | Lepší správa textur, instanced rendering | Nízká úroveň, compute shadery, fronty, paměťové bariéry |
| **Snadnost použití** | Relativně snadná | Stále dosti podobné WebGL | Vyšší složitost – nutné detailní nastavení pipeline |
| **Výkon** | Střední | Vyšší než WebGL | Nejvyšší (efektivnější správa GPU) |
| **Podpora v prohlížečích** | Široká (Chrome, Firefox, Safari, Edge) | Dnes již široká, ale může chybět v některých starších verzích | Postupně přibývá, dosud experimentální v některých prohlížečích |

Z tabulky je zřetelné, že **WebGPU** staví na úplně jiné úrovni přístupu k GPU než WebGL/WebGL2. Zatímco WebGL2 rozšiřuje stávající řešení založené na OpenGL ES, WebGPU vychází z moderních API, která jsou navržena s ohledem na efektivitu a flexibilitu v rámci současného grafického hardware.

Pro **běžné webové 3D aplikace** a pro rychlé prototypování je zřejmě stále nejrozumnější využití **WebGL2** (nebo knihoven nad ním), jelikož dosud nabízí nejširší podporu a spolehlivost napříč prohlížeči.  
Naopak **WebGPU** ocení ti, kdo potřebují skutečně využít potenciál GPU naplno – ať už pro fotorealistické renderování, masivní výpočty, nebo integraci s ML (např. neural nety běžící přímo na GPU v prohlížeči).

### 1.3 Aplikace 3D grafiky na webu

3D grafika na webu není jen zajímavý trend – už dnes se naplno využívá v mnoha oblastech a s vývojem nových technologií se její uplatnění neustále rozšiřuje. Níže uvádím několik hlavních oblastí, kde se 3D řešení uplatňují:

1. **Interaktivní vizualizace**
   * **Vzdělávání**: V různých online kurzech, výukových platformách nebo virtuálních laboratořích najdeme 3D modely molekul, anatomických částí těla či historických artefaktů. Studenti si mohou detailně prohlédnout objekty ze všech stran nebo s nimi dokonce interagovat.
   * **Průmyslové využití**: Zobrazování 3D modelů strojů a zařízení v inženýrských či výrobních firmách. Díky tomu lze urychlit proces návrhu, údržby nebo školení zaměstnanců, kteří se s modelem mohou seznámit předem.
   * **Prezentační a marketingové účely**: 3D grafické prvky dokážou efektivně zaujmout a představit produkt či koncept atraktivním způsobem (např. architektonické návrhy).
2. **3D e-commerce**
   * **Online prohlídky produktů**: Mnohé e-shopy nabízejí 3D modely nábytku, elektroniky nebo třeba bot, takže zákazníci si mohou zboží „osahat“ ze všech stran.
   * **Konfigurátory**: Auta, kola, domácí spotřebiče nebo i oblečení je možné konfigurovat online (vybrat barvu, materiál či doplňky) a ihned vidět výsledek v reálném 3D zobrazení.
   * **Zvýšení důvěry zákazníků**: 3D náhled pomáhá zákazníkům lépe pochopit, co kupují, což může snižovat počet vratek a zvyšovat spokojenost s nákupem.
3. **Hry**
   * **Webové hry s 3D prostředím**: Díky stále rychlejším GPU a pokročilým API jako WebGL2 se webové hry už přibližují kvalitou tradičním desktopovým hrám.
   * **Multiplayer**: Propojení 3D enginů a technologií jako WebSockets nebo WebRTC umožňuje hrát v reálném čase s ostatními hráči přímo v prohlížeči.
   * **Experimenty a prototypy**: Web je ideální platforma pro rychlé sdílení ukázek či dem – nemusí se instalovat žádná aplikace, stačí odkaz.
4. **Virtuální a rozšířená realita (VR/AR)**
   * **WebXR**: Standard umožňující propojit 3D grafiku s VR a AR brýlemi, takže je možné vytvořit například virtuální prohlídky, simulace či hry přímo v prohlížeči.
   * **Rozšířená realita**: S chytrým telefonem lze v prohlížeči zobrazit virtuální objekty přes kameru v reálném prostředí (např. náhled, jak by vypadal nábytek v obývacím pokoji).
   * **Integrace s gesty a ovladači**: Uživatelé se mohou v 3D prostředí volně pohybovat a interagovat s objekty pomocí speciálních ovladačů.
5. **Umělecké projekty a simulace**
   * **3D galerie a generativní umění**: Umělci mohou vystavovat svá díla ve virtuálních prostorách nebo vytvářet interaktivní vizuální instalace.
   * **Hudební vizualizace**: 3D animace reagující na hudební podněty (spektrální analýza, basové linky, rytmy) můžou být atraktivním doplňkem koncertních streamů či DJ setů.
   * **Fyzikální a vědecké simulace**: Například simulace proudění tekutin, modely vesmíru, molekulární dynamiky atd. Přímo v prohlížeči se dají demonstrovat komplexní jevy, a to i s možností interakce (změna parametrů za běhu).

### 1.4 Výhody a omezení

Ačkoli je 3D grafika na webu mocná a slibná, nelze opomenout některé limity, které s sebou přináší. Níže jsou shrnuty hlavní klady a zápory, na které by měl vývojář pamatovat.

1. **Výhody**
   * **Dostupnost a jednoduchá distribuce**: Každý, kdo má moderní prohlížeč, může 3D aplikaci používat ihned, bez nutnosti stahovat a instalovat další software. To výrazně usnadňuje šíření a sdílení – stačí poslat odkaz.
   * **Okamžité aktualizace**: Jakmile je aplikace (nebo hra) hostovaná na serveru, uživatel vždy získává nejnovější verzi. Odpadá tak starost, že někdo používá zastaralou instalaci.
   * **Multiplatformnost**: Web funguje na různých operačních systémech (Windows, macOS, Linux, Android, iOS) i zařízeních (PC, notebook, tablet, mobil). Pokud je projekt navržen správně, uživatelé ho mohou spustit kdekoli.
   * **Komunita a ekosystém**: Existuje široká paleta knihoven (Three.js, Babylon.js, PlayCanvas apod.), návodů a open-source projektů, což zjednodušuje vývoj.
2. **Omezení**
   * **Výkon a kompatibilita**: Výsledná rychlost a kvalita vizualizace závisí na konkrétní kombinaci prohlížeče, zařízení a GPU. Na starších telefonech nebo počítačích nemusí 3D scény běžet plynule, případně se nemusí vůbec načíst.
   * **Komplexita vývoje**: I když existují knihovny, 3D programování obecně vyžaduje znalost matematiky (vektory, matice), shaderů a optimalizačních technik. Pro kvalitní výsledky tak potřebuješ čas a zkušenosti.
   * **Bezpečnostní omezení**: Prohlížeče běží v tzv. sandboxu, aby chránily uživatele i systém. To může některá nízkoúrovňová volání blokovat nebo omezovat (např. přímý přístup k GPU paměti je silně regulován).
   * **Potřeba fallback řešení**: Pokud chceš zajistit podporu i pro uživatele se staršími prohlížeči, musíš myslet na alternativy (např. 2D verzi nebo statické obrázky), pokud prohlížeč nepodporuje WebGL2 či WebGPU.

### 1.5 Stručná historie a vývoj

Historie 3D grafiky na webu je poměrně mladá ve srovnání s tradičním (desktopovým) 3D renderingem. Přesto se během posledních dvou dekád odehrál výrazný posun od experimentálních pluginů k robustním standardům integrovaným přímo v prohlížečích. V této kapitole projdeme hlavní milníky, které umožnily, aby se 3D obsah stal běžnou součástí webových stránek.

* **Počátky (VRML, Flash, Unity Web Player)**
  + **VRML (Virtual Reality Modeling Language)**: Vznikl už v polovině 90. let a byl jedním z prvních pokusů o přenesení 3D grafiky do prostředí webu. Umožňoval popis 3D objektů a scén pomocí textových souborů s příponou \*.wrl. Přestože VRML dokázal do jisté míry zobrazit 3D modely v prohlížeči, nikdy nezískal masivní popularitu. Důvodem byla nízká podpora v mainstreamových prohlížečích a nutnost instalace speciálních pluginů, což komplikovalo uživatelský zážitek.
  + **Flash**: Ačkoli se Flash spojoval primárně s 2D animacemi a interaktivními prvky, existovaly i pokusy o využití 3D (např. papervision3D). Opět se ale jednalo o proprietární technologii vyžadující instalaci pluginu a s limitovaným přístupem k GPU akceleraci.
  + **Unity Web Player**: V době, kdy Unity začalo nabírat na popularitě jako herní engine, nabídl plugin Unity Web Player, který umožňoval spouštět 3D hry a aplikace v prohlížeči. I zde ale platilo, že se jednalo o uzavřenou technologii vyžadující zvláštní instalaci – a s rozšiřováním mobilních zařízení, kde pluginy často nefungovaly, se stávalo toto řešení stále méně životaschopné.
* **Nástup WebGL (2011)**
  + V roce 2011 byla oficiálně vydána první stabilní verze **WebGL 1.0**, která vycházela z OpenGL ES 2.0. Tento krok představoval zásadní zlom, protože WebGL se stal standardem přímo podporovaným prohlížeči (nejprve Chrome a Firefox, později Safari a další).
  + **Hlavní přínosy WebGL 1.0**:
    - Odstranění nutnosti používat pluginy – uživatelé mohli spustit 3D obsah „z krabice“, pokud měli aktuální prohlížeč.
    - Založení na existující technologii (OpenGL ES 2.0) zajistilo relativně rychlý růst a možnost využívat znalosti z herního průmyslu.
  + Po vydání WebGL 1.0 začaly vznikat první populární knihovny jako **Three.js**, které vývojářům usnadnily práci s tímto poněkud nízkoúrovňovým API. Během krátké doby se objevila řada experimentálních dem a her, což dokládalo potenciál 3D grafiky na webu.
* **WebGL2 (2017)**
  + S rostoucími nároky na kvalitu 3D zobrazení a výkon vznikla potřeba posunout WebGL dál. Výsledkem byl standard **WebGL2**, publikovaný konsorciem Kronos Group v roce 2017.
  + **Novinky ve WebGL2**:
    - **Podpora OpenGL ES 3.0** – objevily se funkce jako 3D textury, *transform feedback*, *instanced rendering*, jež usnadnily vykreslování komplexnějších scén a efektů.
    - **Vylepšené shadery a uniformy** – umožňovaly pokročilejší grafické efekty přímo na GPU.
    - **Vyšší stabilita a kompatibilita** – mnozí vývojáři ocenili lepší funkce pro práci s buffery a texturami, což vedlo k efektivnějšímu využití GPU.
  + Přijetí WebGL2 bylo postupné, protože ne všechny prohlížeče (zejména mobilní) ho implementovaly současně a beze zbytku. Postupem času se však WebGL2 stal běžně dostupným v Chrome, Firefoxu či nových verzích Edge (Chromium), zatímco Safari podporu postupně vylepšuje.
* **Příchod WebGPU (2022+)**
  + Nejmladší kapitolu v tomto vývoji představuje **WebGPU**, které se začalo objevovat v experimentální podobě v roce 2020 a do stabilnější fáze se dostává zhruba od roku 2022.
  + **Inspirace moderními API**: WebGPU se opírá o principy a architektury Vulkanu (Khronos), Metalu (Apple) a Direct3D 12 (Microsoft), což jsou nízkoúrovňová API zaměřená na vysoký výkon a detailní kontrolu nad hardwarem.
  + **Rozdíl oproti WebGL/WebGL2**: Zatímco WebGL/WebGL2 jsou postaveny nad OpenGL ES, WebGPU jde cestou přímého, nižšího přístupu k GPU. To poskytuje vyšší efektivitu a flexibilitu při vykreslování, ale také zvyšuje komplexitu implementace.
  + **Možnosti využití**:
    - **Pokročilé grafické efekty** (fotorealistické osvětlení, ray tracing v budoucnu, fluidní simulace).
    - **Strojové učení** (computational shadery dovolují masivní paralelní výpočty, což pomáhá akcelerovat trénování nebo inferenci neuronových sítí).
  + **Současný stav podpory**: WebGPU je v mnoha prohlížečích k dispozici jako experimentální funkce (např. Chrome Canary, Firefox Nightly), je ale jen otázkou času, kdy se stane plnohodnotným standardem se stabilní podporou napříč platformami.

*Vzor:* ***VLASTNÍ TEXT*** *závěrečné práce uspořádaný hierarchicky do kapitol a podkapitol, každá kapitola (úrovně 1) musí být vždy na nové straně.*

# Teoretická část (není název kapitoly)

Autor/autorka uvedou vlastní název kapitoly vztahující se ke konkrétnímu tématu práce

## Porovnání WebGL, WebGL2 a WebGPU

Vývoj 3D grafiky na webu je úzce propojen s technologiemi WebGL, WebGL2 a WebGPU. Každá z nich má jiné možnosti, výkonové charakteristiky, a také různý stupeň podpory mezi prohlížeči. Následující podkapitoly se věnují funkčním rozdílům, podpoře jednotlivých prohlížečů a bezpečnostním hlediskům.

### 2.1 Funkční rozdíly

* **WebGL (OpenGL ES 2.0 pro web)**
  + Základy 3D vykreslování: Poskytuje nízkoúrovňové rozhraní, kde programátor řeší buffery, shadery (vertex/fragment) a transformace.
  + Nižší nároky na znalosti GPU pipeline: V porovnání s novějšími API je WebGL „jednodušší“ a vhodný pro základní 3D scény.
  + Omezená podpora některých pokročilých funkcí: Např. geometry shadery, transform feedback nebo 3D textury zde nebyly nativně dostupné.
* **WebGL2 (OpenGL ES 3.0 pro web)**
  + Rozšířené funkce z OpenGL ES 3.0: Například 3D textury, transform feedback (umožňuje sbírat data z vertex shaderů), instanced rendering (efektivní vykreslování více kopií stejného objektu).
  + Lepší správa paměti a shaderů: Větší flexibilita při práci s různými formáty textur, buffery a větší kontrola nad GPU zdroji.
  + Zpětná kompatibilita: Většina kódu z WebGL 1.0 lze ve WebGL2 používat, ale k dispozici je zároveň více vylepšení a optimalizací.
* **WebGPU (inspirace Vulkan, Metal, Direct3D 12)**
  + Nízkoúrovňový přístup: Dává vývojáři mnohem větší kontrolu nad vykreslovacím řetězcem (pipeline). Je tedy možné optimalizovat výkon na úrovni, která v WebGL/WebGL2 nebyla dostupná.
  + Compute shadery: Umožňují provádět obecné výpočty na GPU, takže WebGPU je ideální nejen pro grafiku, ale i pro strojové učení nebo vědecké simulace.
  + Vyšší výkon a flexibilita: Díky efektivnějšímu využití GPU lze pracovat s masivními 3D scénami, fotorealistickými efekty či složitými paralelními výpočty.
  + Vyšší složitost vývoje: Na rozdíl od WebGL/WebGL2 je nutné spravovat více zdrojů ručně (komandové fronty, paměťové bariéry apod.), což vyžaduje hlubší znalosti grafické pipeline.

### 2.2 Podpora prohlížečů

* **WebGL**: Dnes už v podstatě celoplošně podporované, běží na aktuálních verzích Chrome, Firefoxu, Safari, Edge i většině mobilních prohlížečů.
* **WebGL2**: Stabilní podpora u Chrome, Firefoxu, novějšího Edge (na jádru Chromium), situace v Safari je lepší než dřív, ale i tak může být někdy omezená. Na mnoha mobilních zařízeních funguje, ale setkáš se s různými drobnými omezeními (typ chipsetu, verze OS).
* **WebGPU**: Teprve v ranější fázi adopce. Chrome jej podporuje oficiálně (od verze 113 ve výchozím stavu, dříve za příznakem chrome://flags), ve Firefoxu a Safari je zatím převážně experimentální. Pro mobilní zařízení je to ještě složitější – řada z nich nemá nejnovější API v OS, takže je potřeba další vývoj a optimalizace.

### 2.3 Bezpečnostní aspekty

* **Sandboxing**: Prohlížeče záměrně omezují, co kód s WebGL/WebGPU může dělat, aby chránily hardware a operační systém. V praxi to znamená, že přímý přístup do GPU paměti je filtrovaný a spravovaný prohlížečem.
* **Ověřování shaderů**: WebGL/WebGPU prochází kontrolou kódu shaderů, aby se zabránilo potenciálnímu zneužití zranitelností v ovladačích GPU.
* **Ochrana soukromí**: Přestože GPU dokáže zpracovávat velká data, prohlížeč brání přímému čtení obsahu paměti, ke kterému programátor nemá explicitní přístup. Minimalizuje se tak riziko neoprávněného získání citlivých informací.
* **Potenciální exploit (Spectre/Meltdown)**: Některé bezpečnostní problémy spojené s GPU/CPU (časování operací, spekulativní vykonávání instrukcí) se řeší i v rámci GPU sandboxu; většinou se ale jedná o hlubší úroveň než samotné WebGL/WebGPU API.

## 3. Knihovny pro práci s 3D grafikou

Práce přímo s WebGL či WebGPU je často poměrně nízkoúrovňová – vyžaduje dobrou znalost shaderů, transformací a správy bufferů. Proto vznikly knihovny (frameworky), které vývoj značně usnadňují.

### 3.1 Three.js, Babylon.js, atd.

* **Three.js**
  + Jedna z nejoblíbenějších a nejrozšířenějších knihoven pro 3D na webu.
  + Nabízí ucelený ekosystém: kamery, scény, světla, modely, materiály, animace.
  + Má rozsáhlou komunitu a tisíce pluginů, příkladů i tutoriálů.
  + Je primárně postavena na WebGL (aktuálně WebGL2), experimentální větve už chystají podporu WebGPU.
* **Babylon.js**
  + Další robustní framework s podobným zaměřením.
  + Rychle se vyvíjí a nabízí pokročilé funkce (např. PBR – Physically Based Rendering, podpora WebXR, částečně WebGPU).
  + Má skvělý nástroj Playground, kde lze interaktivně psát kód a rovnou vidět výsledek.
* **Jiné knihovny**
  + **PlayCanvas**: Herní engine s online editorem a zaměřením na výkon.
  + **A-Frame** (nad Three.js): Usnadňuje tvorbu VR/AR aplikací díky HTML-like syntaxi.
  + **Oimo.js, Cannon.js**: Fyzikální enginy pro simulaci pohybu, kolizí a dalších jevů.

### 3.2 Porovnání jednoduchosti práce s těmito knihovnami

* **Three.js**
  + Velmi přehledná a „tradiční“ struktura: vytvoříš scénu, kameru, renderer a poté přidáváš objekty.
  + Dokumentace je rozsáhlá, ale může být pro začátečníka trochu chaotická. Komunita je ovšem obrovská, tutoriály najdeš všude.
  + Pro jednodušší projekty je ideální – stačí pár řádků kódu a zobrazíš první 3D objekt.
  + Nevýhoda: některé pokročilejší efekty (fyzika, post-processing) je potřeba řešit buď doplňkovými knihovnami, nebo custom shaderem.
* **Babylon.js**
  + Má bohatou podporu WebXR a VR/AR. Přináší i editor uzpůsobený pro vytváření scén.
  + Pro úplné nováčky může být Babylon.js trochu robustnější, ale je skvěle zdokumentovaný a Playground usnadňuje experimentování.
  + Jde o „vše v jednom řešení“, do kterého se dají integrovat i pluginy pro fyziku, animace, materiály atd.
  + Disponuje docela aktivní komunitou, i když menší než Three.js.
* **Herní enginy (PlayCanvas, Unity WebGL build)**
  + Poskytují profesionální nástroje pro tvorbu 3D her, včetně editoru a asset pipeline.
  + V některých případech je ale obtížnější propojit tyto enginy s vlastní webovou aplikací (např. Single Page App), oproti integraci menších 3D knihoven.

Obecně platí, že Three.js a Babylon.js jsou skvělé na interaktivní 3D scény, prezentace produktů či menší hry. Pokud je cílem komplexní hra nebo rozšířené VR/AR projekty, lze sáhnout po Babylon.js (kvůli WebXR) nebo některém z herních enginů. V konečném důsledku záleží na preferencích, požadovaných funkcích a osobních zkušenostech vývojáře.

## 4. Parallax Mapping a Šroubovice

### 4.1 Co to je a jak funguje

**Parallax Mapping** (někdy též Parallax Occlusion Mapping – POM) je technika určená k vylepšení vnímání nerovností povrchu materiálu (např. kamene, cihly, dřeva) bez nutnosti používat extrémně hustou geometrii. Využívá tzv. height map (výškovou mapu), která definuje, jak hluboké jsou detaily textury. Při renderování se pak tyto „hloubky“ počítají ve shaderu a simulují efekt nerovností.

* Princip: Místo jednoduchého texturování se do fragment shaderu přidává výpočet průchodu paprsku texturou (stejně jako by byl povrch fyzicky nerovný).
* Výhoda: Relativně malý náklad na GPU oproti reálnému modelování všech detailů v geometrii.
* Nevýhoda: Při extrémních úhlech pohledu může dojít k vizuálním artefaktům (např. „odseknutí“ textury).

**Šroubovice** (Helix) je pak konkrétní 3D útvar, který lze snadno naprogramovat pomocí parametrických rovnic. Zajímavý je, když se k němu aplikuje parallax mapping, jelikož křivky a točivé plochy pěkně demonstrují, jak detailně může vyhlížet povrch.

### 4.2 Využití v 3D grafice

* **Parallax mapping**
  + Herní enginy: Pro vytváření realistických povrchů, např. kamenná podlaha, nerovné cihly na zdi.
  + Architektonické vizualizace: Při pohledu na fasádu či stěny dokáže navodit dojem detailních struktur bez obrovského množství polygonů.
  + WebGL aplikace: Parallax mapping je dobrá ukázka pokročilejších shaderových technik, které lze provozovat v reálném čase i v prohlížeči.
* **Šroubovice**
  + Vizualizace v biologii (DNA helix), fyzice (závitovky, vlnovody), strojírenství (závity na šroubech).
  + Designové efekty: Může jít čistě o estetický prvek, který demonstruje prostorové křivky a materiály.

## 5. Akcelerace AI pomocí WebGPU

S nástupem WebGPU se objevuje příležitost nejen pro lepší grafiku, ale i pro výpočty strojového učení přímo v prohlížeči. Pro některé aplikace (lokální zpracování dat, soukromí, offline režim) je GPU akcelerace na straně klienta velmi užitečná.

### 5.1 Základy strojového učení na grafických procesorech

* **Paralelní výpočty**: GPU disponuje mnoha jádry (ve srovnání s CPU), která mohou zpracovávat tisíce vláken současně. To je ideální pro výpočetně náročné algoritmy, jako je trénování neuronových sítí či masivní maticové operace (např. v convolutional neural networks).
* **Vyšší propustnost**: Pro ML (Machine Learning) úlohy je důležitá rychlá propustnost při práci s velkými maticemi, což GPU dokáže lépe než CPU, které sice může mít rychlejší jádra, ale jich je výrazně méně.
* **Knihovny pro ML**: Existují projekty jako TensorFlow.js, ONNX.js nebo WebDNN, které už využívají GPU akceleraci (dříve jen WebGL, postupně i WebGPU).

### 5.2 WebGPU a jeho výhody pro AI

* **Compute Shadery**: WebGPU umožňuje psát kód, který se spouští přímo na GPU, a nemusí jít o klasické vykreslování trojúhelníků. Díky tomu lze implementovat např. kernel konvoluce pro neurony, redukce, lineární algebra a další ML operace.
* **Menší overhead**: WebGPU lépe pracuje s frontami příkazů a paměťovými zdroji, takže je možné dosáhnout vyššího výkonu než při využití WebGL, který byl původně navržen výhradně pro rendering.
* **Soukromí a lokální výpočty**: Uživatel nemusí posílat citlivá data na server; model lze nahrát do prohlížeče a inference (vyhodnocení) probíhá lokálně. To se hodí např. u aplikací pro zpracování obrazu či zvuku v reálném čase (detekce gest, rozpoznávání hlasu).
* **Předzvěst budoucích standardů**: Jakmile se WebGPU ustálí, je pravděpodobné, že prohlížečové knihovny ML přejdou na WebGPU jako hlavní backend, což by mělo přinést výrazné výkonnostní skoky.

# Praktická část/Empirická část/Vlastní práce (není název kapitoly)

*Autor/autorka uvedou vlastní název kapitoly vztahující se ke konkrétnímu tématu práce*

## Nadpis úrovně 2

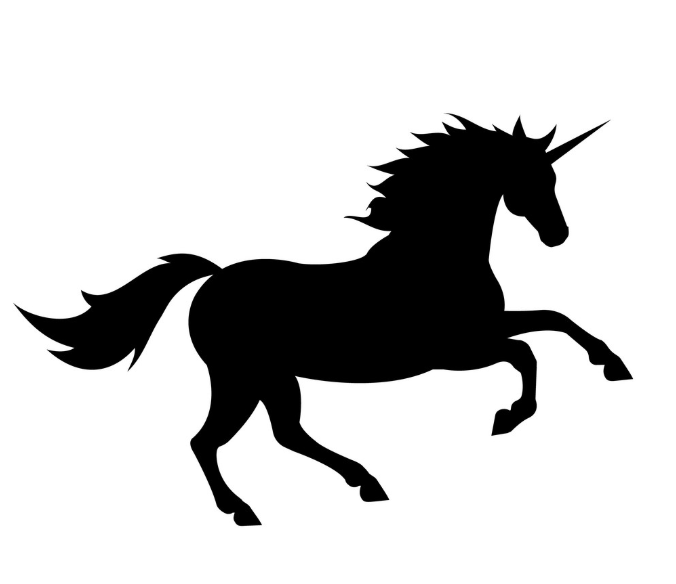
*Obrázek se v textu značí následujícím způsobem: samotný obrázek se označí: „****Obrázek 1: Název obrázku****“ (11 nebo 12 pt, černě, tučně). Obrázky se označují názvem a číslováním nad obrázkem a zdrojovým dokumentem pod obrázkem, příp. informace o vlastním zpracování (11 nebo 12 pt, černě). K popisování doporučujeme využít nástroje textového editoru, který usnadní generování seznamu obrázků na konci práce.*

**Obrázek 1: Logo**



Zdroj: <https://unicornuniversity.net/cs/>

**Obrázek 2: Obrázek jednorožce**



Zdroj: Vlastní zpracování

*Tabulky se označují názvem a číslováním nad tabulkou a zdrojovým textem pod tabulkou. Tabulky, obrázky a grafy se číslují zvlášť. Každá tabulka, obrázek nebo graf MUSÍ být v textu okomentován. Je nepřípustné, aby jednotlivé kapitoly (podkapitoly) tvořilo pouze grafické znázornění v podobě tabulek, grafů, obrázků, schémat atp. bez jejich okomentování.*

**Tabulka 1: Statistika vět zachovaných a vyřazených filtr. kritériem *FK1***

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Sada** | **Celkem** | **Zachováno** | **Vyřazeno** | **Zachováno** |
| **dtest** | 5228 | 2384 | 2844 | 45,6 % |
| **etest** | 5476 | 2419 | 3057 | 44,2 % |
| **train-1** | 4709 | 2204 | 2505 | 46,8 % |

Zdroj: Vlastní zpracování

**Matematické rovnice, vzorce**

*Pokud jsou v práci rovnice, nezapomeňte je správně číslovat. Pro jejich zápis používejte MS Editor rovnic, případně jinou obdobnou aplikaci. Rovnice by měla vypadat například takto (nezapomeňte proměnné popisovat):*

|  |  |
| --- | --- |
|  | (1) |

*kde S je obsah kruhu o poloměru r .*

## Vykreslení Trojúhelníku s Přechodem RGB

### Vytvoření ukázkových projektů v Three.js, WebGL2 a WebGPU

### Srovnání výkonu (FPS) a složitosti kódu

## Vytvoření Hello World v Různých Technologiích

### Stejně jako výše, ale s jednoduchým "Hello World" příkladem

## Komponenta v Canvas v uu5

### Vývoj komponenty v uu5

### Vypsání statistik, výkonu

## Šroubovice s Paralax Mappingem

### Implementace šroubovice s paralax mappingem

### Analýza a srovnání výkonu

## Experimenty s AI a Grafikou

### (Nad rámec) Experimenty s přímým přístupem ke grafice pro strojové učení

# Závěr

Tato kapitola se nečísluje. Rozsah je zpravidla 5-10 normostran.

# Seznam použitých zdrojů

V seznamu zdrojů musí být uvedeny všechny v závěrečné práci citované zdroje. Zároveň nesmí seznam obsahovat zdroje, které nejsou v závěrečné práci použity.

Používáme citační normu ČNS ISO 690. Doporučujeme pro tvorbu citací některý z citačních nástrojů, které jsou v základní verzi zpravidla zdarma dostupné.

# Seznam obrázků (existují-li)

Obrázek 1: Logo 11

Obrázek 2: Obrázek jednorožce 12

**Seznam tabulek (existují-li)**

[Tabulka 1: Statistika vět zachovaných a vyřazených filtr. kritériem *FK1* 12](#_bookmark9)

# Seznam grafů (existují-li)

# Seznam příloh (existují-li)

*Každá příloha musí být alespoň jednou odkázána do vlastního textu práce. Přílohy se číslují. Každá příloha začíná na nové stránce.*

# Příloha A – Název přílohy

# Příloha B – Název přílohy