Popis Projektu:

Engine pro Textově založené hry *(Text Based Game Engine)* je Aplikace založená v Eclipse, která slouží k hraní textových her vytvořené komunitou. Zároveň jejím uživatelům bude přístupný systém k vytváření těhto příběhů. Postup v hraní příběhu může být uložen a znovu načten.

Příběhy budou čteny ze souborů *.nodes*, které lze uživateli snadno upravit. Příběhy můžou mít větvící se dialog, bojové sekce nebo dokonce I sběratelné předměty a příkazy které budou využívat API enginu *(Přidat zdraví, resetovat příběh, uložit, přidat předmět etc.),* ty jsou volané přímo ze souborů .nodes.

Popis Ovládání:

Aplikace bude ovládána pomocí standartního grafického rozhranní.

Návrh Programování:

Aplikace bude číst a zapisovat všechna data do JSON souborů. Soubory .nodes budou obsahovat data příběhu (dialog, možnosti, příkazy, bojové sekce, předměty etc.) v syntaxu JSON. Pomocí GSON Knihovny lze rychle iterovat mezi objekty a JSON soubory.   
Pro zobrazování bude použito standartní SWT/JFRAME rozhranní s možnými úpravami. Pro případné hostování příběhů může být použit jakýkoliv adresář dostupný pod adresou a používající specifické mapování složek a souborů.

Co je hotovo:

* GUI Vypisování a vstup -, nyní plně responzivní
* Základní systém pro pracování s příběhy
* Update pro přehledný a modulární management při práci s kódem, vše je take dokumentováno v JavaDoc, pokročilý popis v prezentaci.
* Základní řízení pro StoryNode a BattleNode
* Multithreading a zlepšený výkon v Loopech a vláknech
* Ukládání, Příkazy a tzn. Tokeny (viz Prezentace)
* Celý changelog viz [gitHub](https://github.com/Brokop/Text-Adventure)

Co teď nefunguje:

* Rozklíčování GUI vstupu a převedení do node ID nebo popř příkazu, systém proto po vstupu nic neudělá a bude čekat na vstup který odpovídá indexu ID.
* **Systém na načítání uložených tokenů, problem A: Ukládají se I duplikáty ID, B: Nenačítají se pomocí GetCondition();**

Současný Systém

Při zapnutí se vytvoří instance enginu která běží async s GUI vláknem.   
Během načítání GUI se pomocí **Init()** v Engine vláknu zkontrolují a načtou potřebné soubory k spuštění příběhu, popř se vytvoří USERSAVE.sav nebo se načte existující do S\_Data.

Po dokončení načítání GUI a Příběhových *.nodes*, převezme kontrolu nad vypisováním a vstupem, nastavení příběhu samotného (stále v Engine vláknu asynchroně s GUI).

Vstup se nyní provádí univerzálně pomocí **input2id()** metody, která vrací **id** odpovědi.

Vytváření Příběhů

Všechny data příběhů jsou uloženy v **“Buňkách”** v souboru pod příponou .*nodes*. Tyto soubory se rozlišují podle jména na:

**DIALOG.nodes** – *Obsahuje data potřebná pro průběh příběhu: Dialogy, Možnosti, Eventy, Rozcestí, Příkazy etc.*

**BATTLE.nodes** – *Obsahuje data pro “bojové sekvence”,*

**ITEMS.nodes** – *Obsahuje počátky systému na zakládání objektů.*

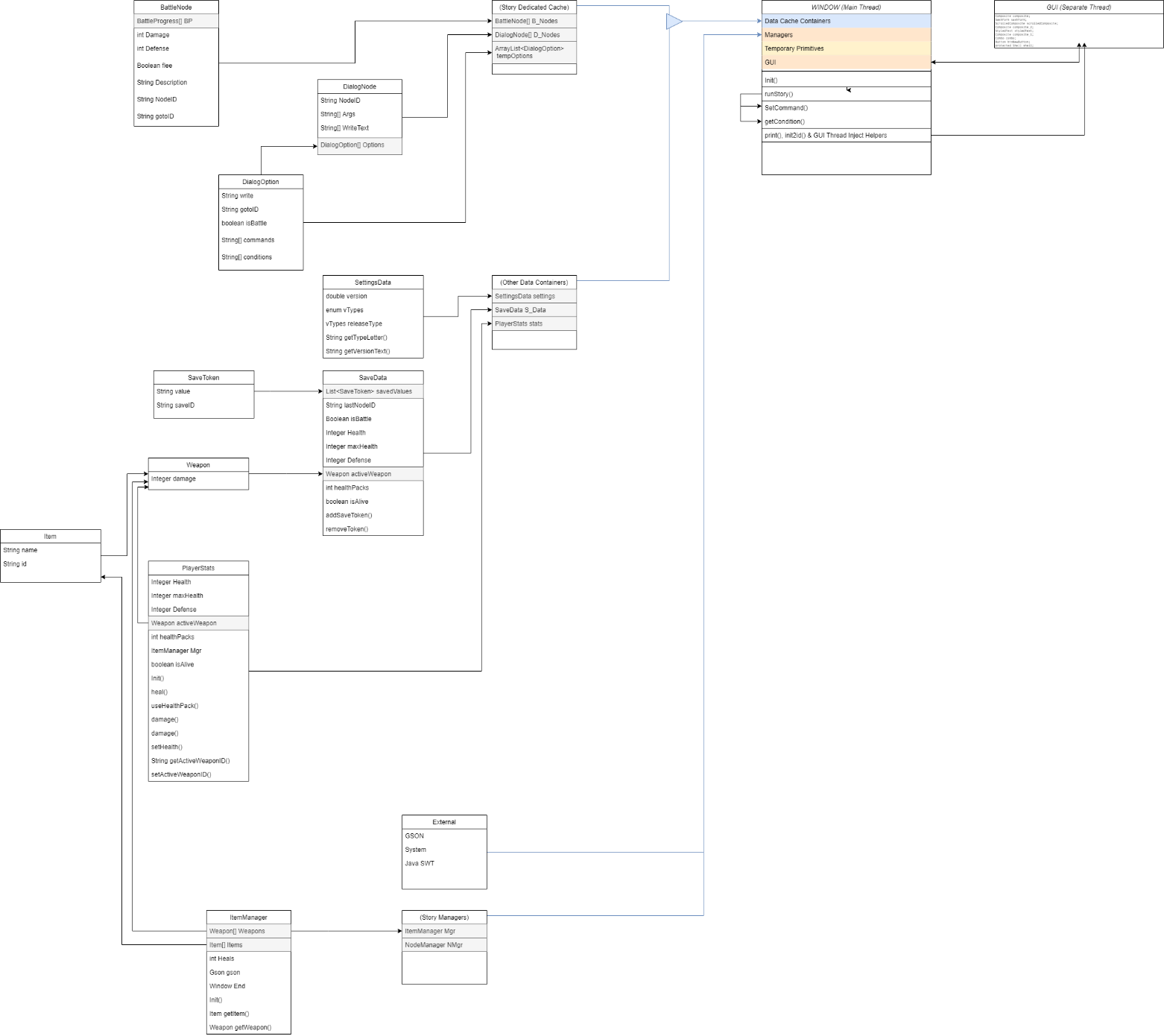
**SETTING.json** – *Bude obsahovat data nastavení aplikace, zatím jen verze.*

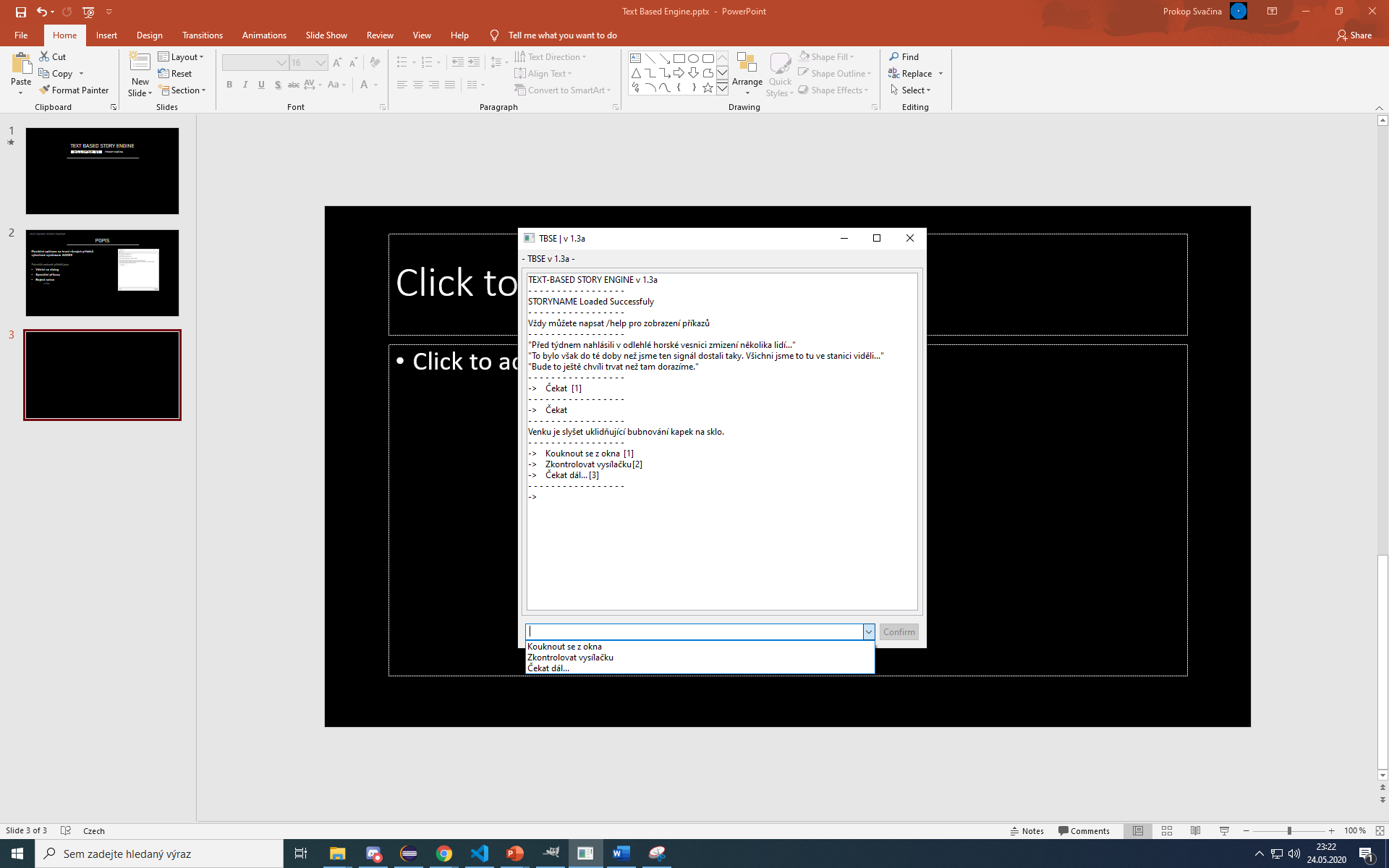
**Pro plnou dokumentaci API Enginu je k dispozici JavaDoc nebo dodatečné kometáře ve zrojovém kódu. Nyní je I k dispozici powerpoint prezentace, která stručně vysvětluje vyváření příběhů, funkcí etc.**

Současný Souhrn:

Místo toho abych dělat demonstrační příběh co ukáže všechny současné funkce TBSE sem se soustředil na ledění ukládání Tokenů a Načítání jich jako podmínky. Bohužel jsem ani jeden z těhto nových nápadů na poslední chvíli nestihnul zrealizovat a celkověmě to zbrzdilo. Pokud budete chtít můžu popřípadě dale pracovat a ladit tento projekt k jeho plnohodnotnému potenciálu.

* *Svačina*

****

**Popis GUI (viz Prezentace)**

Potvrzení Vstupu

Možnosti příběhu

Hlavní Vstup

Vypisování pravděpodobných možností vstupu