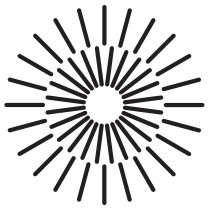
FAKULTA MECHATRONIKY,
INFORMATIKY A MEZIOBOROVÝCH
STUDIÍ TUL



Projekt

Tvorba módu pro hru Minecraft

Studijní program: B0613A140005 – Informační technologie
Studijní obor: Aplikovaná informatika
Autor práce: **Jiří Růta**
Vedoucí práce: Ing. Jana Vitvarová

Liberec 2024

Tento list nahraďte
originálem zadání.

Prohlášení

Prohlašuji, že svůj projekt jsem vypracoval samostatně jako původní dílo s použitím uvedené literatury a na základě konzultací s vedoucím mého projektu a konzultantem.
Jsem si vědom toho, že na můj projekt se plně vztahuje zákon č. 121/2000 Sb., o právu autorském, zejména § 60 – školní dílo.
Beru na vědomí, že Technická univerzita v Liberci nezasahuje do mých autorských práv užitím mého projektu pro vnitřní potřebu Technické univerzity v Liberci.
Užiji-li projekt nebo poskytnu-li licenci k jeho využití, jsem si vědom povinnosti informovat o této skutečnosti Technickou univerzitu v Liberci; v tomto případě má Technická univerzita v Liberci právo ode mne požadovat úhradu nákladů, které vynaložila na vytvoření díla, až do jejich skutečné výše.
Beru na vědomí, že můj projekt bude zveřejněn Technickou univerzitou v Liberci v souladu s § 47b zákona č. 111/1998 Sb., o vysokých školách a o změně a doplnění dalších zákonů (zákon o vysokých školách), ve znění pozdějších předpisů.
Jsem si vědom následků, které podle zákona o vysokých školách mohou vyplývat z porušení tohoto prohlášení.

26. 5. 2024 Jiří Růta

Poděkování

Tímto bych rád poděkoval vedoucí svého bakalářského projektu Ing. Janě Vitvarové za podporu při zpracování této práce.

Tvorba módu pro hru Minecraft

Abstrakt

Mód je software, který upravuje nebo rozšiřuje jiný již existující software. Módy jsou často sdíleny jako otevřený software. V tomto bakalářském projektu se zaměřuji na vytváření módu pro hru Minecraft napsaný v programovacím jazyku Java.
Vytvořený mód přidává síta, kterými může hráč přesívat tekutiny na rudy, lis který drtí předměty na tekutiny a kotlík ve kterém může hráč kombinovat tekutiny a předměty. Mód je kompatibilní s ostatními populárními módy.
Výsledkem práce je vydání módu na stránku curseforge.com.

Klíčová slova: Minecraft, mód, Java, Otevřený software

Creating a mod for Minecraft

Abstract

Mod is a software that is used to modify different already existing software. Mods are often shares as a open source software. This bachelor project focuses on rocess of creating a mod for game Minecraft written in programing language Java .
Created mod adds sieves, which can player use to sift liquids into ores, a press that crushes items into liquids and a cauldron in which the player can combine liquids and items. The mod is compatible with other popular mods.
The result of this work is the release of the mod on curseforge.com.

Keywords: Minecraft, modding, Java, open source

Contents

List of Figures

List of Tables

1 Minecraft

Ve hře Minecraft se celý svět skládá z bloků. Hráč tyto bloky může vykopat a poté je položit, anebo z nich vyrobit vybavení.

Na začátku hry hráč vykope dřevo. Poté otevře uživatelské rozhraní svého inventář. Zde jsou všechny předměty, která má hráč u sebe. Kromě toho je zde uživatelské rozhraní ve kterém hráč může vyrábět předměty. Hráč si vyrobí dřevěnou sekeru, která mu umožní kopat dřeva rychleji.

Hráč může vykopat každý blok ve hře. Vykopaný blok se mu poté přidá do inventáře. Doba kopání závisí na druhu bloku a na vybavení hráče.

Hráčův inventář obsahuje všechny předměty, které má u sebe. Hráč si může otevřít uživatelské rozhraní svého inventáře kdykoli chce. Inventář ve hře Minecraft je rozdělen na políčka. Každé políčko může obsahovat pouze jeden druh předmětu a pouze omezené množství.

Součástí uživatelské rozhraní hráčova inventáře je i uživatelské rozhraní pro výrobu nových předmětů. Hráč v něm může vyrábět nové vybavení anebo bloky z věcí v jeho inventáři. Některé bloky mohou hráči povolit vyrobit si věci, které bez nich nemohl. Například pec umožní hráči předělat surovou rudu na ingoty.

2 Minecraft z programátorské stránky

2.1 Blok

Minecraft server musí mít v paměti minimálně 1382400 bloků pro každého hráče připojeného na serveru. Každý blok je v paměti skladován jako odkaz na jeho objekt v registru bloků a odkaz na jeho *blockstate*.[1] V registru bloků jsou uchovávané všechny neměnné vlastnosti a v *blockstate* uchovává všechny vlastnosti které se pro blok mohou měnit.

Při startu hry se pro každý druh bloku zaregistruje objekt třídy *Block* do registru. Tento objekt je neměnný. Veškeré instance daného bloku odkazují na tento jeden objekt. Jelikož je tento objekt společný pro všechny instance, pokud chceme ukládat data specifická pro jeden blok musíme použít *blockstate* anebo *tile entity*.

Blockstate je neměnná třída obsahující pouze proměnné. Pro každou kombinaci proměnných je vygenerován singleton. Každý blok odkazuje na jeden ze singletonů. *Blockstate* šetří paměť, tím že dovoluje několika blokům držet pouze jednu proměnou (odkaz na *blockstate*) místo několika. Například máme blok lampy. Každý blok lampy odkazuje na stejný objekt v registru bloků, ale pokud chceme určit zdali je lampa zapnutá musíme se podívat na který *blockstate* odkazuje.

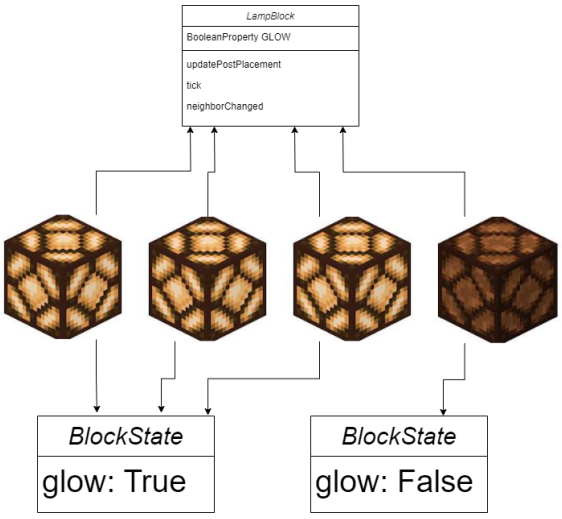


Figure 2.1: Příklad *blockstate*

Pokud potřebujeme v bloku skladovat velké množství proměnných, můžeme využít *tile entity*. Objekt třídy *tile entity* je samostatný pro každý blok. Například

blok bedna umožňuje hráči odložit si předměty. Které předměty jsou uloženy v bedně je uloženo v *tile entity*. Bedna také odkazuje na blockstate, který udává kterým směrem je otočena. A každá bedna také odkazuje na svůj objekt v registru bloků.

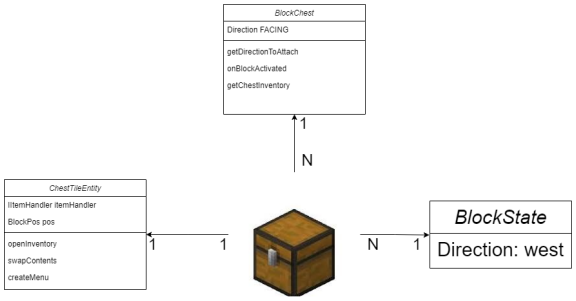


Figure 2.2: Příklad vztahů

2.2 Předměty a recepty

Inventář ve hře Minecraft je rozdělen na políčka. Každé políčko může obsahovat pouze jeden druh předmětu a pouze omezené množství. V programu je každé políčko reprezentováno objektem *itemstack*.

Třída *itemstack* obsahuje 2 důležité proměnné, jaký předmět obsahuje (odkaz na jeho třídu) a jeho množství. Třída obsahuje převážně metody na porovnávání objektů *Itemstack* mezi sebou, předávání informací o předmětu, ukládání a načítání dat.

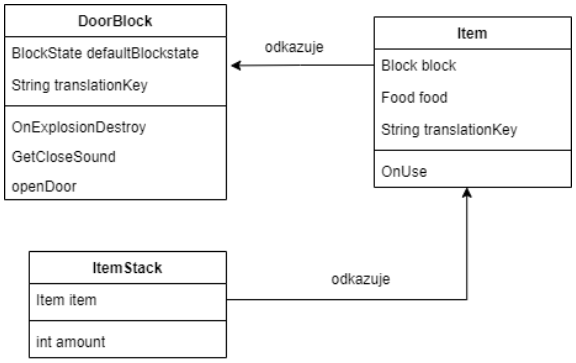


Figure 2.3: ItemStack vztahy

Každý druh předmětu má svůj singleton, který dědí třídu *Item*. Ten je registrovaný do registru předmětů při spuštění hry. Tato třída obsahuje proměnné a metody pro daný předmět. Například obsahuje metodu "onUse", která je volána pokud se hráč pokusí použít předmět. Jelikož je tato třída společná pro všechny předměty pokud chceme ukládat data specifická pro jeden předmět musíme použít *NBT*. *Named binary tag*, zkráceně *NBT*, je systém pro ukládání dat jako pár klíče a hodnoty.

Velká část hry Minecraft je vytváření nových předmětů z těch které už hráč vlastní. Z čeho se předměty vyrábějí je zapsáno v JSON souborech ve složce *resources/data/id/ recipes*.

Ve hře je několik druhů receptů, každý druh receptu má svojí třídu implementující rozhraní *IRecipie*. Každý JSON soubor ve složce s recepty má hodnotu "type", která říká kterou třídou ho má hra přečíst. Při zapnutí se načtou veškeré recepty z JSON souborů a pro každý JSON soubor se vytvoří jeden objekt třídy implementující *IRecipie*. Tyto objekty obsahují veškerá data z JSON souborů a implementují metody na porovnání zdali je možné recept vyrobit.

2.3 Klient a Server

Kód hry Minecraft vždy běží dvakrát, jednou jako server a jednou jako klient. I když hráč hraje sám na vlastním počítači, Minecraft pořád běží rozdělen na stranu serveru a klienta. Na obou stranách běží stejný kód. Na rozlišení se používá metoda "*world.isRemote*", která vrací *true* pokud kód běží na klientu. Některé operace musí běžet pouze na jedné straně a druhou stranu upozornit pomocí paketů.

Například vytváření nových předmětů se musí provádět na serveru, jinak je možné že server nebude o předmětu vědět a když se ho hráč pokusí použít tak nebude fungovat, protože o něm server neví. Na druhou stranu, vykreslování grafiky se musí provádět pouze na klientu, jelikož server nemá soubory potřebné k vykreslení grafiky, pokus o vykreslování na serveru vyústí v chybu a pád programu na straně serveru.

Většina módu je dělaná, aby běžela na klientu a serveru, ale je možné udělat mód co běží pouze na jedné straně. V takovém případě stačí aby byl mód nainstalovaný na straně na které běží. Samozřejmě mód který běží jenom na serveru, anebo jenom na klientu je omezený v tom co může dělat. Na kterých stranách musí být mód nainstalován je napsáno v souboru *resources/META-INF/mods.toml*.

Některé bloky nebo předměty otevřou hráči uživatelské rozhraní. Například když hráč klikne na pec, otevře se mu uživatelské rozhraní ve kterém může odložit předměty do pece, aby je pec předělala na jiné. Uživatelské rozhraní se ve hře Minecraft skládá ze dvou částí: *Container* a *Screen*.

Container obstarává logiku a data. Komunikuje s ostatními částmi hry. Běží na serveru i na klientu.

Screen určuje jak bude uživatelské rozhraní vypadat. Zobrazuje text a obrázky. Komunikuje pouze s *Container*. Může obsahovat logiku, ale měla by být jen okolo grafiky v uživatelském rozhraní. Běží pouze na klientu.[2]

3 Vytváření modu

Minecraft v základu nepovolujeme módování, pokud chceme vytvořit mód musíme použít zavaděč módů, který se pustí při startu hry a načte kód módu do hry za pomoci *Java Class Loader*. Zavaděč umožňuje přidat nové objekty do příslušných registrů, kde s nimi může komunikovat základní hra i veškeré módy. Zavaděče módu také poskytují vývojářům módů události, které jsou volány během hry a vývojáři na ně mohou reagovat.Existují 2 zavaděče módu *Forge* a *Fabric*. Tyto zavaděče nemohou být spuštěny společně.[3]

Forge se zaměřuje na kompatibilitu mezi módy. *Forge* má větší množství událostí a jsou k němu zabalené abstraktní třídy a rozhraní pro většinu věcí, které vývojáři módu často implementují. Díky tomu mají vývojáři normalizované rozhraní, která mohou používat ke komunikaci s ostatními módy. *Forge* je také starší, takže většina módů je dělaná pro *Forge*.

Fabric se zaměřuje na výkon. *Fabric* má menší API a na rozdíl od *Forge* načítá pouze kód který je potřeba. *Fabric* také dovoluje upravovat soubory základní hry.

Z pohledu hráče jsou zavaděče pouze omezení, které omezuje jaké módy hráč může využívat společně.

Pro svůj mód jsem si vybral *Forge*, jelikož se v něm jednodušeji řeší kompatibilita mezi ostatními módy a více módu je dělaných pro *Forge*.

Hra Minecraft vyšla v roce 2009, od té doby vyšlo 20 velkých aktualizací. Hráči mohou hrát jakoukoli verzi. Jelikož se s každou verzí hry Minecraft se mění kód základní hry, musí se pro každou verzi udělat nová verze zavaděče módu. Módy napsané pro jednu verzi hry Minecraft nejsou kompatibilní s ostatními verzemi. Proto se při verzování módu udává hned za verzí módu verze hry Minecraft pro kterou je napsán. Například mód *Ex Aqua 0.4 - 1.16.5* je mód pro verzi 1.16.5 hry Minecraft a verze samotného módu je 0.4 . Některé módy jsou dělané na více verzí hry Minecraft a jejich autor je pravidelně předělává na nové verze. Když je mód hodně populární, ale autor ho nechce předělat na ostatní verze hry Minecraft, často se najde jiný programátor který udělá podobný mód na jiné verze, proto pro staré populární módy je často několik neoficiálních pokračování.

Před vytvářením módu je potřeba se rozhodnout pro jakou verzi hry Minecraft a pro jaký zavaděč bude programátor mód vyvíjet. Oba zavaděče nabízejí šablonu na stažení. Šablona obsahuje *Gradle* soubory, které stáhnou všechny soubory potřebné pro zavaděč a také celý zdrojový kód vybrané verze hry Minecraft.

Minecraft je produkt společnosti Microsoft. Microsoft dovoluje používat zdrojové kódy jejich hry pro vývoj módů, dokud vývojář šíří pouze svůj mód a nešíří zdrojové kódy hry Minecraft. Vývojář je vlastníkem svého kódu, ale nesmí mód prodávat.

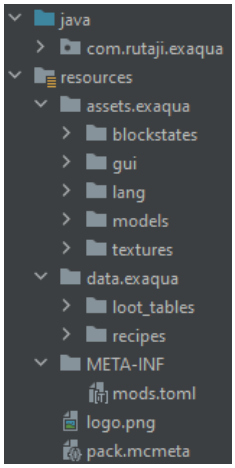


Figure 3.1: Hierarchie souborů

Může žádat o peněžní dary. Oba zavaděče nijak právně neomezují vývojáře.

Po přípravě šablony si vývojář musí zvolit id pro svůj mód. Id je textový řetězec, který nesmí obsahovat speciální znaky kromě _“. Tento textový řetězec bude použit jako část cesty k souborům módu a také pro identifikaci módu.

Všechny módy dodržují stejnou hierarchii souborů. Mód je rozdělen na složku *java* a složku *resources*. Pod složkou *java* se nachází složka se jménem autora a pod ní složka pojmenovaná jako id módu, v ní se nachází veškerý Java kód. Ve složce *resources* jsou ostatní soubory rozděleny na *assets* a *data*. Složka *assets* obsahuje soubory, které by měli být potřebné pouze na klientu, zatímco složka *data* obsahuje soubory potřebné na klientu a serveru. Ve složkách *assets* a *data* se vždy nachází složka pojmenovaná po id módu.

Ve složce *resources/assets/id/blocksates* se nachází JSON soubory, které říkají jaký model má být pro blok použit na základě jeho *blockstate* proměnných. Pro každý blok je jeden tento soubor. Jméno tohoto souboru odpovídá jménu pod kterým je blok registrován v registru bloků. Ve složce *resources/assets/id/models* jsou JSON soubory obsahují modely pro bloky a předměty. Soubor také obsahuje odkazy na textury použité v modelu. Textury se nacházejí ve složce *resources/assets/id/textures*. Ve složce *resources/assets/id/lang* je JSON soubor pro každý jazyk, obsahující klíč a jeho překlad. Ve složce *resources/assets/id/gui* je uložena grafika využitá v uživatelském rozhraní.

Ve složce *resources/data/id/recipes* se nacházejí JSON soubory receptů. Každý soubor obsahuje předměty potřebné jako vstup a předměty které budou vytvořeny jako výstup receptu. Ve složce *resources/data/id/loot_tables* jsou uložené JSON soubory ze kterých se načítá jaký předmět by měl být vytvořeny při různých situacích, jako je zničení bloku nebo zabití zvířete. Fungují hodně podobně jako recepty, ale používají se v jiných případech.

V souboru *resources/META-INF/mods.toml* jsou informace o módu jako jeho jméno, verze, popis a autor. Hlavně v něm je seznam závislostí a pro jakou verzi hry Minecraft je mód napsaný. Je v něm napsáno zdali musí být mód nainstalován na straně klienta a serveru. [2]

4 Můj mód z pohledu hráče

Módy často přidávají bloky které představují různé stroje, jako třeba elektrická pec anebo Hydraulický lis. Tyto bloky většinou mají vlastní inventář do kterého vloží hráč předměty a stroj z nich vyrobí jiné. Tyto stroje většinou potřebují elektřinu, aby fungovali, což je věc která se v základní hře nevyskytuje a je exkluzivní pouze pro módy.

Těmto módům se říká technické módy. Dobrý technický mód funguje s ostatními technickými módy. Například pokud mód přidává generátor tak mělo být možno z generátoru odvádět elektřinu i kabely z cizího módu a napájet s ní stroje z cizích módů.

Můj mód přidává několik strojů. Přidává síta, lis a kotlík. Síto přesívá tekutiny na rudy. Lis drtí předměty na tekutiny. V kotlíku může hráč kombinovat tekutiny a předměty, aby získal nové tekutiny. Všechny fungují s trubkami a kabely z populárních technických módů. Také obsahuje přímou podporu pro populární mód JEI. Mód je volně dostupný pro stažení na stránce *curseforge.com* a jeho zdrojový kód je ve veřejném GitHub repositáři pod GPL licencí.

Můj mód je inspirován módem *Ex Nihilo*. Jedná se o populárním mód vydaný v roce 2014. Tento mód přidává síta ve která přesívají hlínu na rudy a sud ve kterém může hráč kombinovat tekutiny a předměty, aby získal nové tekutiny. Tyto bloky nepotřebují elektřinu.

Tento mód je neudržovaný a funguje pouze pro velmi starou verzi hry Minecraft, ale má několik neoficiálních pokračování.

Ex Nihilo 2 je jediné oficiální pokračování k módu *Ex Nihilo* od jeho autora. Bylo vydáno roku 2015, ale od roku 2016 je opuštěné. Obsahově je podobné původnímu *Ex Nihilo*, ale některé věci chybí, mód zůstal nedodělán.

Ex Nihilo Omnia je neoficiální pokračování vydané 4.9. 2016. Tento mód přidává vše co původní *Ex Nihilo* pouze s malými úpravami a obsahem navíc. Tento mód dostal svoji poslední aktualizaci v roce 2019. *Ex Nihilo Adscensio* je neoficiální pokračování vydané 29.11 2016. Tento mód je první který přidal více úrovní sít. Poslední aktualizaci dostal 6.5 2017.

Ex Nihilo Creatio je fork *Ex Nihilo Adscensio*. Je to první mód který přidává sivky, které hráč musí napájet elektřinou, ale za odměnu vyrábějí bez hráčovi přítomnosti. Mód byl vydán 14.8. 2017 a poslední aktualizace byla vydána 27.8 2019.

Ex Nihilo: Sequentia začalo jako *Ex Nihilo Creatio* předělané na vyšší verze hry Minecraft. Tento mód obsahuje veškerý obsah ze všech předchozích *Ex Nihilo* inspirovaných módů. Byl vydán 4.8. 2020 a je stále podporován. Má několik módů které dodávají další obsah k tomuto módu anebo přidávají kompatibilitu s ostatními

Table 4.1: Seznam podobných módů

jméno módu	podporované minecraft verze	zavaděč	počet stažení
Ex Nihilo	1.6 a 1.7	Forge	8 058 815
Ex Nihilo 2	1.8	Forge	61 152
Ex Nihilo Omnia	1.10 a 1.12	Forge	2 611 291
Ex Nihilo Adscensio	1.10	Forge	6 236 591
Ex Nihilo: Creatio	1.12	Forge	15 638 553
Ex Nihilo: Sequentia	1.15 až 1.20 (nejnovější)	Forge	6 761 281
Fabricae Ex Nihilo	1.18 až 1.20 (nejnovější)	Fabric	168 881
Ex Aqua	1.16	Forge	432

módy.

Fabricae Ex Nihilo neoficiální pokračování módu *Ex Nihilo* pro zavaděč Fabric. Vydáno 21.3. 2022 a stále aktualizováno.

Dalším populárním módem který by měl každým dobrý mód podporovat je JEI. Tento mód přidává uživatelské rozhraní , ve kterém může hráč procházet seznam všech předmětů ve hře, zobrazovat si proces jejich výroby a jejich využití. Jedná se o nejpopulárnější mód s více jak 300 miliony stažení.

Tyto data jsou z ze stránky *curseforge.com* a z GitHub repositářů vybraných módů. [4] [5]

5 Můj mód z pohledu programátora

Při programování síta jsem nejdříve vytvořil třídu *SieveBlock*, která dědí ze třídy *Block*. Tato třída obsahuje neměnná data a také propojuje blok s jeho *tile entity* (*SieveTileEntity*) a uživatelským rozhraní. Tuto třídu jsem zaregistroval do registru bloků.

Při registraci vyžaduje každý blok unikátní jméno. Toto jméno je pak použito k vyhledání modelu a textur ve složce *resources*. Model pro síto a ostatní bloky jsem vytvořil v modelovacím programu Blockbench a textury jsem maloval v programu Aseprite.

Třída *SieveTileEntity* je implementace *tile entity* pro blok síta.Obsahuje data unikátní pro každé síto, jako předměty, tekutiny a elektřinu v sítu. Pro tyto data jsem napsal metody k ukládání a načítání dat z NBT *(named binary tags)*.

Tile entity také spravuje přesívání tekutin na předměty v sítu. Každý herní cyklus zkontroluje zdali má dostatek tekutin a energie na přesívání a pokud ano tak zmenší množství elektřiny a tekutiny a přidá do svého inventáře předměty podle receptů.

Recepty jsou JSON soubory udávající výsledky výroby a její podmínky. Pro jejich čtení a práci s nimi jsem napsal samostatnou třídu *SieveRecipe*.Jedna instance této třídy představuje jeden recept. Tato třída čte data z JSON souborů, porovnává inventář zdali vyhovuje receptu, čte a zapisuje data do paketů. Ke každému druhu receptu jsem napsal třídu, která překládá daný recept pro JEI mód a přidá mu uživatelské rozhraní.

Skladování tekutin obstarává třída *MyLiquidTank* která dědí ze třídy *FluidTank* z Forge knihovny. Implementoval jsem všechny její metody a přidal pár vlastních abych zabránil duplikaci kódu. Aby mohli s touto třídou komunikovat i ostatní cizí módy vytvořil jsem třídu *WaterFluidTankCapabilityAdapter* která slouží jako adaptér. Tato třída překládá mojí třídu *MyLiquidTank* na *IFluidTank* což je rozhraní zabudované v Forge. Toto rozhraní je používané většinou módů pro komunikaci. Testoval jsem tento kód s nejznámějšími módy a se všemi fungoval.

Jelikož potřebuji synchronizovat *MyLiquidTank* mezi server a klientem vytvořil jsem si třídu *MyFluidStackPacket* která je poslána ze serveru klientovy a obsahuje momentální stav daného *MyLiquidTank*. Odesílání a přijímaní tohoto paketu je implementováno ve třídě MyLiquidTank.

Skladování elektřiny je podobně vyřešeno ve třídě *MyEnergyStorage* která dědí z *EnergyStorage* z Forge knihovny. Pro komunikaci s ostatními módy jsem vytvořil třídu *EnergyStorageAdapter* který překládá *MyEnergyStorage* na IEnergyStorage z Forge knihovny. Synchronizace mezi server a klientem je řešena odesíláním a

přijímáním třídy MyEnergyPacket z *MyEnergyStorage*.

SieveContainer je třída která obstarává logickou část uživatelského rozhraní. Slouží jako most mezi *SieveTileEntity* a *SieveScreen*. Tato třída umožňuje hráči interagovat s inventářem síta. Bere informace přímo z *SieveTileEntity* a předává je pro *SieveScreen*. *SieveScreen* je poté vykreslí do uživatelského rozhraní. Podobným postupem jsem naprogramoval i ostatní předměty v módu.

Seznam použité literatury

1. COOPER, J.W. *Java Design Patterns: A Tutorial*. Addison-Wesley, 2000. ISBN 9780201485394. Available also from: https://books.google.cz/books?id=SrJRu8T69FcC.
2. *forge documentation* [online]. [visited on 2024-05-21]. Available from: https://docs.minecraftforge.net/en/1.16.x/.
3. WATSON, Nicholas. *Re-Crafting Games: The inner life of Minecraft modding*. 2019. PhD thesis. Concordia University.
4. *curseforge* [online]. [visited on 2024-05-21]. Available from: https://www.curseforge.com/minecraft/mc-mods.
5. *github* [online]. [visited on 2024-05-21]. Available from: http:https://github.com/.