05 Přidání textury a nastavení itemu

Nastavení textury

Aktuálně náš item nemá žádnou texturu. Ukážeme si tedy, jak texturu přidat.

Ukázkovou texturu si můžete stáhnout zde.

> **Stažení souboru**
>
> Na odkaz je potřeba kliknout pravým tlačítek myši a soubor stáhnout. Levým tlačítkem myši se obrázek pouze otevře.

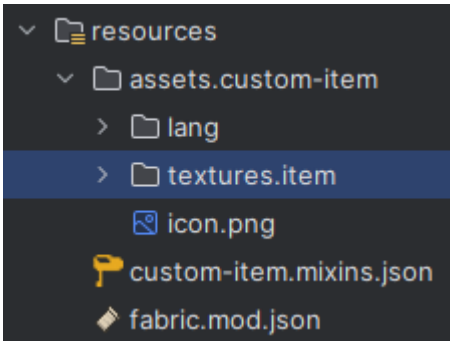
Pro přidání textury potřebujeme přidat 2 soubory - texturu a popis toho, jak se má textura zobrazovat (model itemu). Tyto soubory se přidávají do složky `resources/assets/<id modu>`.

- `<id modu>` je v mém případě `custom-item`

Přidání textury

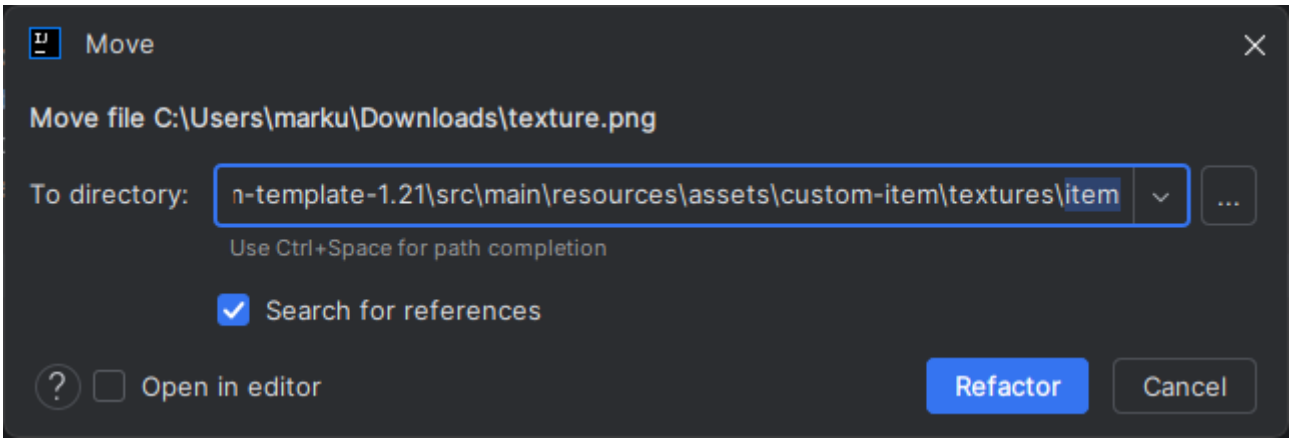
Textury se přidávají do složky `textures`. V této složce jsou poté další složky podle toho, pro co je textura určena - pro item je to složka `item`. Vytvoříme si tedy složku `textures/item`.

Struktura složek by měla vypadat následovně:



Do této složky si přesuneme staženou texturu a soubor přejmenujeme podle názvu našeho itemu. V mém případě tedy `magic_dust.png`.

Pokud soubor přesouváme přetažením myši, tak se nám zobrazí následující dialog:



Zde stačí zvolit možnost *Refactor*.

> **Přejmenování souboru**
>
> Na soubor klikneme pravým tlačítkem myši a zvolíme *Refactor* > *Rename*

Model itemu

Dále musíme hře říct, že má tuto texturu použít pro náš item a jak přesně se má zobrazovat. Na to musíme vytvořit `json` soubor, kterým určíme model itemu.

Stejně jako u textury si i na modely musíme vytvořit složku - `models`. A v této složce jsou opět složky podle toho, pro co je model určený. Vytvoříme si tedy složku `models/item`.

V této složce si vytvoříme soubor, který opět pojmenujeme podle našeho itemu. Bude to soubor typu `json`, takže na konec názvu souboru musíme uvést i tuto příponu. V mém případě tedy `magic_dust.json`.

> **Vytvoření souboru**
>
> Soubor vytvoříme tak, že klikneme pravým tlačítkem myši na složku a zvolíme *New* > *File*

Základní model itemu vypadá následovně:

```
{
  "parent": "item/generated",
  "textures": {
    "layer0": "custom-item:item/magic_dust"
  }
}
```

`parent`

určuje, z jakého modelu vycházíme. Ve hře už jsou nějaké základní modely vytvořené - např. právě pro itemy, které jsou jen obyčejný obrázek. Díky tomu nemusíme nastavovat, jak přesně bude item otočený, jak bude velký a podobně. Stačí nám tedy jen určit obrázek, který se má použít.

`textures`

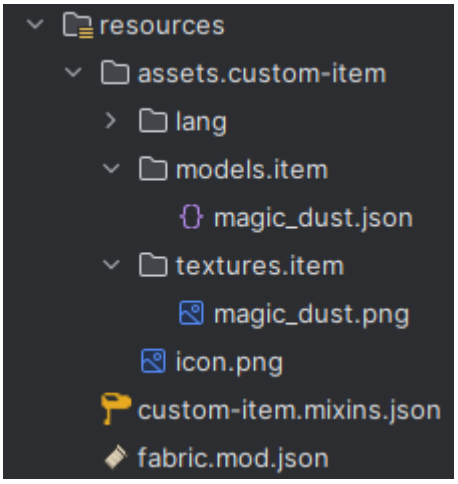
zde definujeme, jaké obrázky se mají pro zobrazení itemu použít. Textury můžeme skládat i z více vrstev, ale nyní nám bude stačit jen jedna vrstva a to je právě vlastnost `layer0`.

- zde opět `custom-item` je id módu a `item/magic_dust` je cesta k obrázku

> **Zajímavost**
>
> V programování se většinou začíná číslovat od 0 a ne od 1, jak jsme zvyklí. Proto je první vsrtva označená `layer0`

Když nyní spustíme hru, tak by náš itemu už měl mít texturu.

Výsledná struktura souborů a složek by měla vypadat následovně:



Velikost stacku

Dále si ukážeme, jak změnit počet itemů, co můžeme mít maximálně v jednom slotu/stacku.

To změníme ve třídě `ModItems`, kde jsme si přidávali náš item.

Při vytváření itemu můžeme pomocí dalších metod změnit základní nastavení itemu. Například metodou `maxCount` můžeme změnit maximální počet itemů v jednom slotu.

```
1   public static final Item MAGIC_DUST = register(
2                           new Item(new Item.Settings().maxCount(16)),
3                           "magic_dust"
4   );
```

Upozornění

Pokud nastavíte číslo **menší než 0** nebo **vyšší než 99**, tak hra nemusí fungovat správně. V Minecarftu se ve verzi 1.21 se u itemu zobrazuje maximální číslo 99, ale je možné nastavit i vyšší číslo.

Doporučení

Při nastavování maximálního počtu itemů je dobré se držet hodnot používaných ve hře. Tedy hodnoty `1`, `16` a `64`

Kompostování itemu

U itemu dále můžeme nastavit třeba i to, že jej můžeme přidat do kompostu. To se dělá podobně jako u nastavení, že je item možné použít jako palivo.

Na toto se v Minecraftu opět používají registry. Tentokrát je to `CompostingChanceRegistry`. V metodě `initialize` ve třídě `ModItems` si tedy přidáme řádek, kterým náš item přidáme do tohoto registru.

Stejně jako u přidání paliva i nyní použijeme metodu `add`. Tentokrát jako druhý argument uvádíme, jaká je šance, že se do kompostu přidá další vrstva. Tato hodnota je desetinné číslo mezi hodnotou `0` a `1`, kde 1 je 100% (vrstva se přidá vždy). U desetinného čísla musíme ještě přidat písmeno `f`, abychom označili, že se jedná právě o desetinné číslo (datový typ `float`).

Pokud chceme, aby šance na přidání další vrstvy byly 30%, tak použijeme hodnotu `0.3f`.

```
1   public static void initialize() {
2           ItemGroupEvents.modifyEntriesEvent(ItemGroups.INGREDIENTS)
3                                   .register((itemGroup) ->
    itemGroup.add(ModItems.MAGIC_DUST));
4
5           FuelRegistry.INSTANCE.add(ModItems.MAGIC_DUST, 5 * 20);
6           CompostingChanceRegistry.INSTANCE.add(ModItems.MAGIC_DUST,
    0.3f);
7   }
```

Poznámka

Použitím čísla vyšší než 1 se nic nestane a hra se bude chovat stejně, jako bychom nastavili hodnotu na 1. Ale obecně doporučuji se držet daného rozsahu hodnot.

Celá třída `ModItem` by aktuálně měla vypadat následovně (změněné řádky jsou zvýrazněné):

```
public class ModItems {
    public static final Item MAGIC_DUST = register(
            new Item(new Item.Settings().maxCount(16)),
            "magic_dust"
    );

    public static void initialize() {
        ItemGroupEvents.modifyEntriesEvent(ItemGroups.INGREDIENTS)
                .register((itemGroup) -> itemGroup.add(ModItems.MAGIC_DUST));

        FuelRegistry.INSTANCE.add(ModItems.MAGIC_DUST, 5 * 20);
        CompostingChanceRegistry.INSTANCE.add(ModItems.MAGIC_DUST, 0.3f);
    }

    public static Item register(Item item, String id) {
        Identifier itemID = Identifier.of(CustomItem.MOD_ID, id);
        Item registeredItem = Registry.register(Registries.ITEM, itemID, item);
        return registeredItem;
    }
}
```