

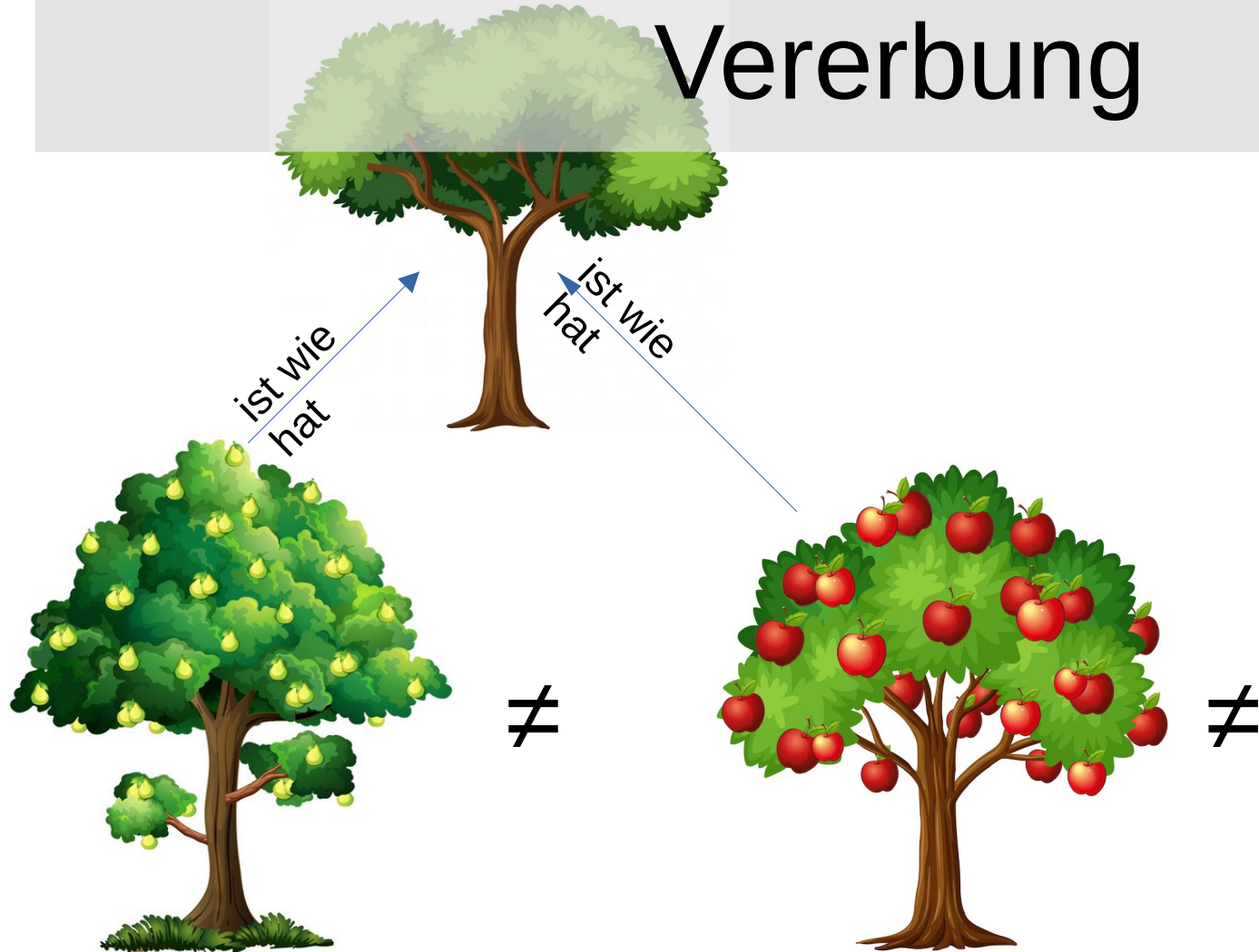
Tools

Eigene Werkzeuge craften

Eigenes Werkzeug

- Neue eigene Item-Klasse, die von Item erbt
 - Klasse anlegen, erweitern, verwenden
- Collections
 - Listen, Maps und Sets
- Loops
 - for, while, foreach
- Idee: der Stummel soll Blöcke zerschlagen
 - Heuballen zu 9 Weizen
 - getrocknete Seetangblöcke in 9 getrockneten Seetang

Vererbung



Vererbung bei Klassen

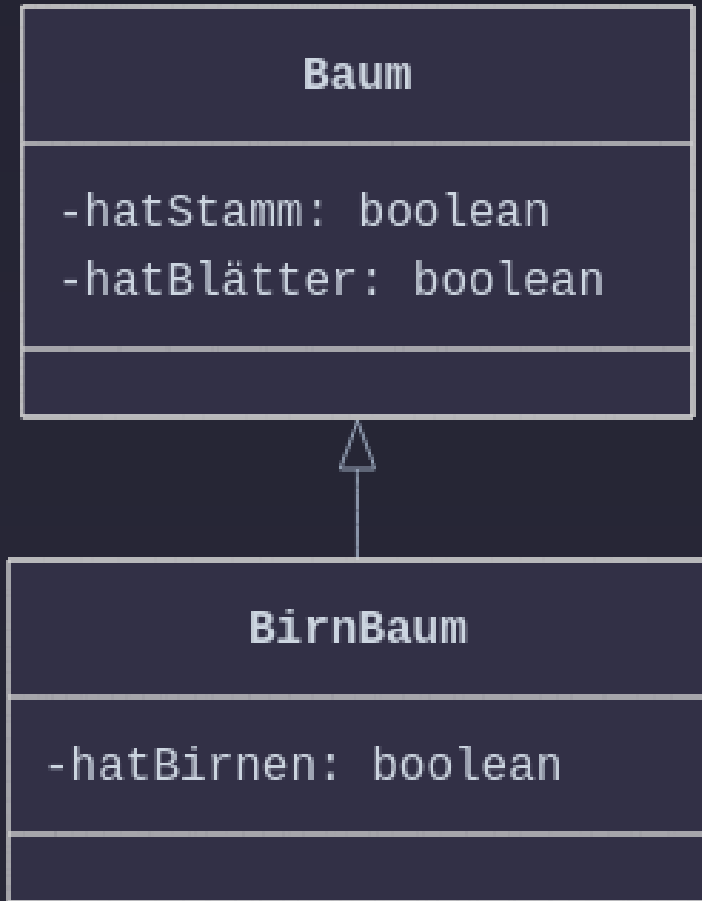
- Klasse Baum
 - Hat einen Stamm – lässt vor dem Winter die Blätter fallen
 - Hat Blätter – verfärbt sich im Herbst
- Klasse Apfelbaum erbt von Klasse Baum
 - Hat Äpfel
- Klasse Birnbaum erbt von Klasse Baum
 - Hat Birnen
- Klasse Bananenstaude erbt nicht
 - Hat Bananen – hat einen Strunk



Beispiel Vererbung im Java-Code

```
public class Baum {  
    private boolean hatStamm = true;  
    private boolean hatBlätter = true;  
}
```

```
public class BirnBaum extends Baum {  
    private boolean hatBirnen = true;  
}
```



Stummel-Klasse angelegt

- Neben die Klasse ModItems: StubItem

```
public class StubItem extends Item {  
    public StubItem(Properties properties) {  
        super(properties);  
    }  
}
```

Sammlungen (Collections)

- Mehrere gleichartige Elemente wie in einem Korb
 - Eier in Eierpackung
 - Limo in Limokiste
- Verschiedene Collection-Typen
 - Listen (gleiche Elemente)
 - Maps (benannte Elemente in einer Liste)
 - Sets (Listen ohne Duplikate)



Änderungen in Map registrieren

- Map<Block, Item> als Container für Blöcke, die man mit dem Stummel zerschlagen kann

```
private static final Map<Block, Item> SLASH_MAP = Map.of(  
    Blocks.HAY_BLOCK, Items.WHEAT,  
    Blocks.DRIED_KELP_BLOCK, Items.DRIED_KELP);
```

Loops

- Wiederholte Ausführung

- Zähle rückwärts von 10 bis 1

- Idee: $10 - 1 = 9$, $9 - 1 = 8$, $8 - 1 = 7$, ... bis $2 - 1 = 1$
 - Als Algorithmus: $X = 10$. $X = X - 1$. Solange machen, wie $X \geq 1$
 - `for (int x = 10; x >= 1; x--) { ausführen }`
 - Alternative: while-Schleife (solange machen, wie gilt: $X \geq 1$)
 - `int x = 10; while (x >= 1) { x--; ausführen }`

- Collections und Loops

- Foreach (für alle Elemente in der Liste / dem Set / der Map)
 - `List<Integer> list; for (Integer i : list) { ausführen }`

Programmieridee

- der Stummel soll Blöcke zerschlagen
 - Heuballen zu 9 Weizen
 - getrocknete Seetangblöcke in 9 getrockneten Seetang
- Verwenden der for-Schleife (1 – 9 Items pro Block)
- Verwenden der Map (Ersetzen des richtigen Items pro Block)

Das Zerschlagen programmieren

```
@Override
public InteractionResult useOn(UseOnContext context) {
    Level level = context.getLevel();
    BlockPos pos = context.getClickedPos();
    Block block = level.getBlockState(pos).getBlock();
    if (SLASH_MAP.containsKey(block) && !level.isClientSide()) {
        level.setBlockAndUpdate(pos, Blocks.AIR.defaultBlockState());
        level.playSound(null, pos, SoundEvents.HOE_TILL.value(), SoundSource.BLOCKS);
        for (int i = 0; i < 9; i++) {
            ItemStack stack = new ItemStack(SLASH_MAP.get(block));
            float scatter = new Random().nextFloat();
            ItemEntity entity = new ItemEntity(level, pos.getX(), pos.getY(), pos.getZ(),
                stack, scatter, scatter, 0.2);
            level.addFreshEntity(entity);
        }
    }
    return InteractionResult.SUCCESS;
}
```

Item-Registrierung anpassen

- Nicht mehr von item, sondern von StubItem
- In registerItem einen zweiten Parameter verwenden, der die StubItem instanziiert
- **StubItem::new** legt eine neue Klasse von StubItem an, sobald sie benötigt wird

```
final Item STUB = registerItem("stub", StubItem::new);
```

Arbeit sichern

- Git commit und push