Amethyst Algorithms Project 1: Coding Basics

Henry Post

Drakonix Systems, LLC

March 31, 2025

About the author

Henry Post

```
bio  -> meltingscales.github.io
code -> github.com/meltingscales/amethyst-algorithms
```

This presentation was made using Markdown and Pandoc with the “beamer” template.

What is this?

Amethyst Algorithms is a course aimed at smart highschoolers or average college students, meant to teach you how to do software programming using Minecraft and NeoForged.

If you're age 15+, like Minecraft, and want to learn programming, you should give this course a try.

Tooling setup

You'll need to install a few tools first!

If using Windows, I highly recommend using Chocolatey `choco` to install packages. It is a fantastic package manager similar to `apt` or `yum`.

- choco

If using Linux, this will depend. I recommend Ubuntu for beginners.

1. Install IntelliJ IDEA
   - Windows JetBrains Toolbox
   - Linux/OSX JetBrains Toolbox
2. Install Git
3. Install Java Development Kit 21
   - Windows Java 21
   - Linux/OSX Java 21

Git setup

I'll assume you've installed Git. You just need to run the below: 1. Open a terminal and `cd` to the place you'd like the repository to live. 2. Run `git clone https://github.com/meltingscales/amethyst-algorithms-minecraft-intro-to-cs-course`

Java setup

Nothing to be done! Just install JDK 21 (not JRE 21).

IDE setup

Next, we're going to set up your IDE.

(Live demo: Henry shows you how to set it up and import)

Basics of Java code: Syntax 1

Let's go to `/src/main/java/io/meltingscales/amethystalgorithms/DirtDropsGold.java` in our editor!

Java syntax can be broken up into a few distinct types. We'll cover them in the next slide.

- `L13`: an annotation! `@EventBusSubscriber(...)`
- `L14`: a class declaration! `public class DirtDropsGold {...}`.
- `L16`: A method (also function) declaration! `onBlockBreak(...){...}`
- `L27`: A variable declaration! `int blockX = blockPos.getX();`
- z: An `if` statement!
- a
- b
- c

Basics of Java code: Syntax 2: Annotations

Annotations are ways to store extra data (“metadata”, literally “data about data”) about a method (function), variable, or class.

In NeoForged, they are very powerful and mean specific things. They tell the (classloader?) to hook specific functions into your mod.

```
@EventBusSubscriber(modid = AmethystAlgorithms.MODID, bus = EventBusSubscriber.Bus
public class DirtDropsGold {

  @SubscribeEvent(receiveCanceled = true)
  public static void onBlockBreak(BlockEvent.BreakEvent event) {
    // get broken block position
     //
  }
}
```

Basics of Java code: Syntax 3: Class declarations

Basics of Java code: Syntax 4: Function/method declarationsx

Basics of Java code: Syntax 5: Variable declarations

Basics of Java code: Syntax 6: `if` statements

Basics of Java code: Syntax 7: `boolean` expressions

Basics of Java code: Syntax 8: method/function invocations

Basics of Java code: Syntax 9: “method chaining”

It looks pretty neat. Method chaining is a useful technique that lets you easily modify the state of an object.

It works because the method just repeatedly returns `this`, meaning the object that is currently running the method.

Basics of Java code: Variables

Basics of Java code: Mod Annotations

Basics of Java code: Syntax