Benjamin Northrop

CS5004

4/10/24

Spring 2024

**Woodshop Game**

I found inspiration for this originally in Lab 3 – the automated house factory. I have a woodworking shop in my garage – it’s a combination of tools from my Dad and things I’ve picked up over time. It’s a hobby, however, due to family obligations and work, I don’t have much leftover time to spend in there. As a result I’ve fruitlessly searched for “woodworking games” that I could dabble in for a few minutes here and there.

This project takes inspiration from the idea of being able to use each tool as an object and work on other objects. It primarily works with the idea of your “workshop” being the model of the program – it holds all of the internal details, inventories, etc., held on the left side of the attached UML diagram. This workshop is composed of several collections – you have your tool collection, a jig1 collection, your wood collection, your built furniture collection, and your selection of blueprints to build from.

The premise is, you go to work to get enough money to “buy” or unlock items and make them available to you in your collection. You can buy wood, tools, or jigs. Wood comes in two forms – it is either plywood or dimensional wood2. Purchasing a tool will “unlock” it and allow you to use it to cut wood. Wood is considered a “cuttable object”

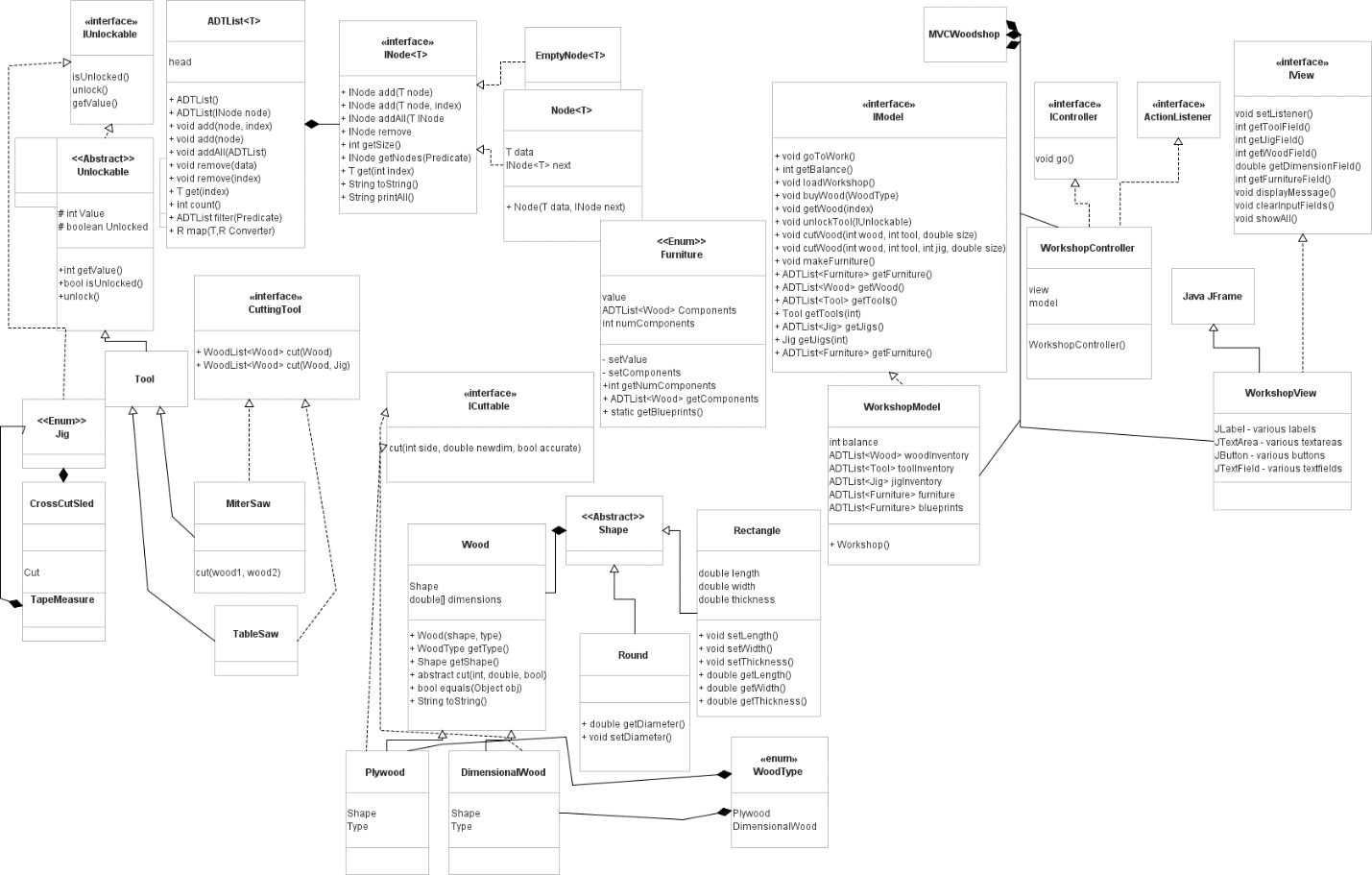
Tool have certain properties that determine their purpose. Table saws are meant for rip3 cuts, and as such it will always cut “parallel” to the longer dimension of the piece of wood we are cutting. (When speaking of dimensions, we’re only going to refer to the length and width for now, as thickness is going to be a set dimension for this application). Miter saws are meant for “cross cuts”4 and can only cut parallel to the shorter dimension, and they have a maximum reach of 12”. Therefore, we cannot cut across a piece of plywood (48” wide) until it’s been cut down into smaller strips on the tablesaw.

Jigs allow us to do some change. A cross-cut sled used with a tablesaw allows us to do cross-cuts on it, therefore we can cut either dimension of a piece of wood. The application will default to the crosscut when a jig is used, and a rip cut when no jig is used.

*Note: A feature that has not yet been incorporated, but the groundwork has been laid out, is the miter saw has an accuracy factor. It does not have a built in measuring system like the tablesaw, so a miter saw would introduce a small variation to the desired dimension (due to eyeballing the distance) until it is used in conjunction with a tape measure, which would provide accurate cuts.*

After cutting wood to the correct dimensions, we can built furniture. We choose the furniture we want to build, and the application will compare our inventory to the components required in the furniture item chosen, and only create the furniture if all components are there. This saves it in our inventory and removes the wood from our inventory.

*Note: future capability would be the ability to sell furniture to get money, as each piece of furniture has an associated value.*



**Concept Map**

|  |  |
| --- | --- |
| **Concept 1: Recursion in Practice** | Recursion is used in each of the ADTList instances inside the model. The primary one that gets manipulated is the wood inventory list. We primarily use this to cut wood, initiated by the model by with the method cutWood() (**WorkshopModel.java Line 146).** This method takes the inputs from the controller, searches for the appropriate tool, jig, and wood in its respective ADTList, then passes those parameters to the tool for the correct procedure. That tool then passes it along to the wood, which actually does the transformation. The code for the actual get recursion is in **ADTList.java Line 76, Node.java Line 29, EmptyNode.Java Line 134.**  It is also used in the makeFurniture (**WorkshopModel.java line 187)** process, in that once we receive the required information from the controller, the model will determine what piece is being made, verify the required pieces are in the inventory, then iterate through the blueprints’ components **(Furniture.java)** to systematically remove the wood pieces from the inventory. This removal combines the recursive get method with an equality check to ensure it only removes the correct piece. |
| **Concept 2: Logical Structure/Design using Abstract Classes and Interfaces** | This design heavily incorporates interfaces and abstract classes to allow for expansion and different variations in the tools. Tools are broken down into its abstract class and |
| **Concept 3: Useful/Logical Abstractions using Generics and Lambda Expressions** |  |
| **Concept 4: Higher Order Functions** |  |
| **Concept 5: Hierarchical Data Representation or Linked List ADT** |  |
| **Concept 6: MVC Design Pattern** |  |
| **Concept 7: SOLID Design Principles** |  |

Glossary:

1jig – an item that works in harmony with a tool to either allow it to do something it wasn’t originally designed for, or to improve the performance of the tool.

2Plywood / Dimensional – Plywood are thin sheets of wood glued together in alternating grain patterns. It provides strength, rigidity, and stability at low cost. Dimensional wood is a single piece of solid wood, typically with a dimensional measurement like 2x4. It looks better and is easier to work with in some manners.

3rip cuts are where you cut parallel to the grain direction, so you can make long, narrow strips. Typically this coincides with the long dimension of a piece of wood.

4cross cuts are when you cut perpendicular to the grain. This typically cuts along the shorter dimension of wood.