Peter McGill

Software Solution Explanation

I'll explain the solution in the order I put the project together. Since I knew I wanted to make an object to serve as a representation for each component, the first file I made was Leaf.java. I knew parsing would give the bounds in a string form, so I used the split method to put each of the numbers into an array, so it would be easier when I went to annotate. Speaking of parsing, the second file I worked on was XMLParsing.java. The hasChildNodes method was especially helpful here because it was how I was able to determine where the leaf components were, and add them to the List. I also surrounded everything in a try catch, because I knew one of the xml files was broken, so I wanted the program to still work for the others. I did the same thing with ImageUpdater.java. This is the file where I made use of the coordinates method in Leaf.java, because I knew rectangles would be the easiest way to put the highlights on the png files. The last file I put together was the Main.java file. This one gave me the most trouble, because it took me a while to figure out how to match the png and xml files, without letting a DS_Store file get in the way (which I learned about during testing). My solution to this was to have two variables to track the most recent xml and png file while iterating (since the matching ones were always together), and then only call the other files if the variables matched, and the current iteration was on one of them (to prevent updating the same file twice when the DS_Store file showed up). All of this contributed to the final result.