MicroRTS

Checkpoint 3

Toolchain

Source code management tool - Github

Build systems - Apache Netbeans

Documentation tool - Doxygen will be used

IDEs - Apache Netbeans

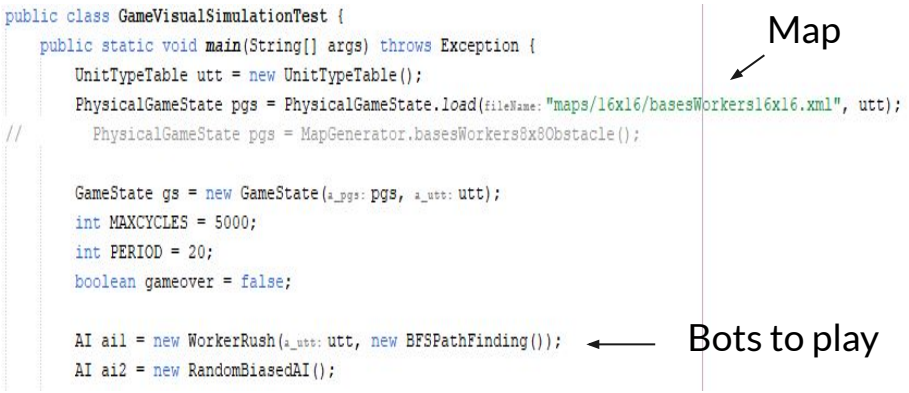
Static analysis tools - Apache Netbeans has a built-in feature for static analysis for Java

Testing tools - Apache Netbeans

Any other tools/libraries/SDKs/framework instrumental in your project - VS Code for editing code, MICRORTS library by Farama Foundation

Toolchain - Netbeans and Visualization

To test a bot, MicroRTS has a file GameVisualSimulationTest.java. This class allows to play any bot against each other on a given map. We use Netbeans, the Java IDE to test our bots since the tutorial for setup for MicroRTS was done on Netbeans and the setup is relatively complex. Netbeans also offers static analysis built into the IDE.



Progress

We each implemented our own bot based on the prototyping, we did on the original MINS bot. We created separate branches and began development on our own bots and have begun to test them against each other and the default bots available (ie. Worker Rush).

Miguel: I implemented a bot that attempts to combine the ideas of proxy rush and light rush. The idea behind my strategy is to quickly establish a barracks unit close to enemy structures and try to quickly mobilize a large number of light units to overwhelm the opposition before they can establish a baseline for resources and others. The intent is to quickly outleverage the opponent. However, the strategy mainly hinges upon successful establishment of the initial barracks and a strong and effective initial attack with the light units. Should that fail, my bot would be likely to lose. Different test scenarios will be noted as they are conducted.

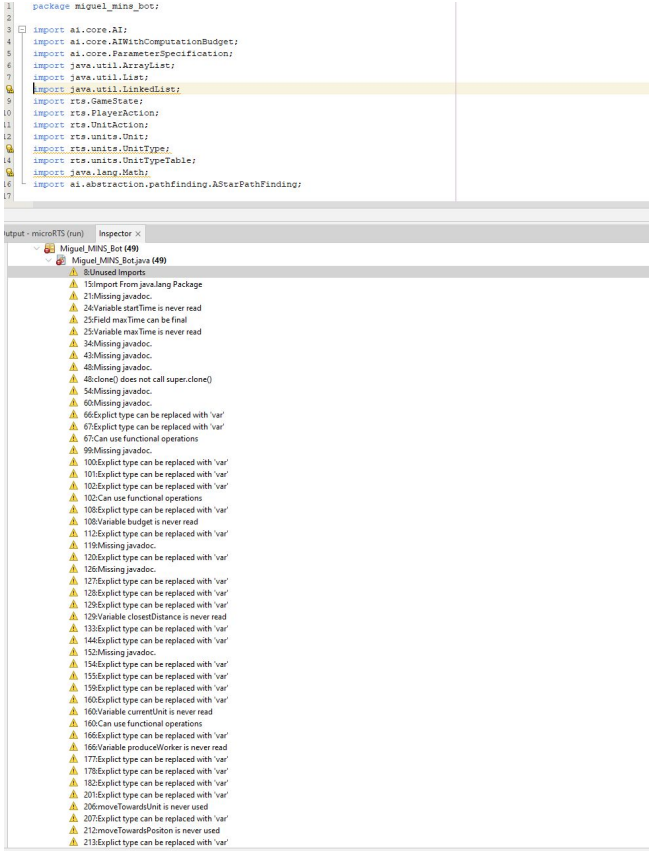
Progress

Matthew: I further developed the AI and experimented with several methods for improvement. The primary addition I made was creating a method for basing some decisions on simulations of potential future games while others remain hard coded. This allows units to perform actions based fully on simulated games, perform fully hardcoded actions, and perform action simulated to be the best from a limited set of actions expected to be beneficial. I also attempted several methods to increase performance including more efficient tracking of changes to the game state during a turn.

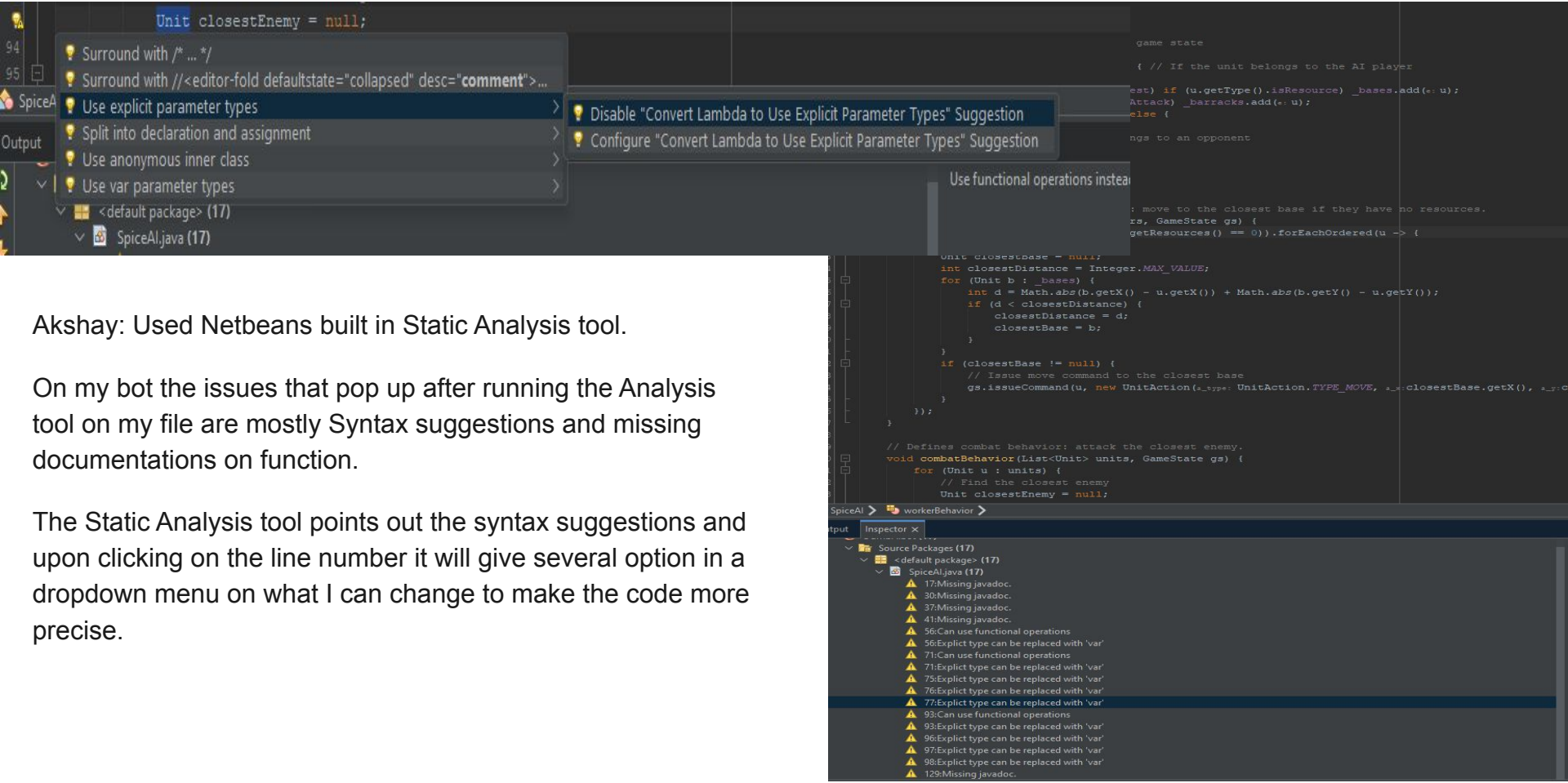
Akshay: I developed a bot in a separate branch which is based on our initial code of MIPS bot and mayari bot, this bot will will be a defensive/ resource gathering bot to began the match and will primarily use light and heavy units to attack enemy barracks and flank to enemy base. I will be basing the pathing of the Units from the mayari bot code to make them work in as many maps as I can without the units bugging out and running into terrain and doing nothing.

Static Analysis

Miguel: I used Netbeans for my static analysis. Almost all issues are suggestions on how to improve the code, such as changing explicit types to just ‘var’. There was one problem where I was missing a type for a variable, but other than that, there is nothing that would critically affect the security or performance of the project. I fixed the missing variable type issue. There are also a number of “never used” or “not read” issues that stem from me not yet having used or planning to use these methods. The methods I don’t plan on using I will delete to clean up the code.



Static Analysis



Akshay: Used Netbeans built in Static Analysis tool.

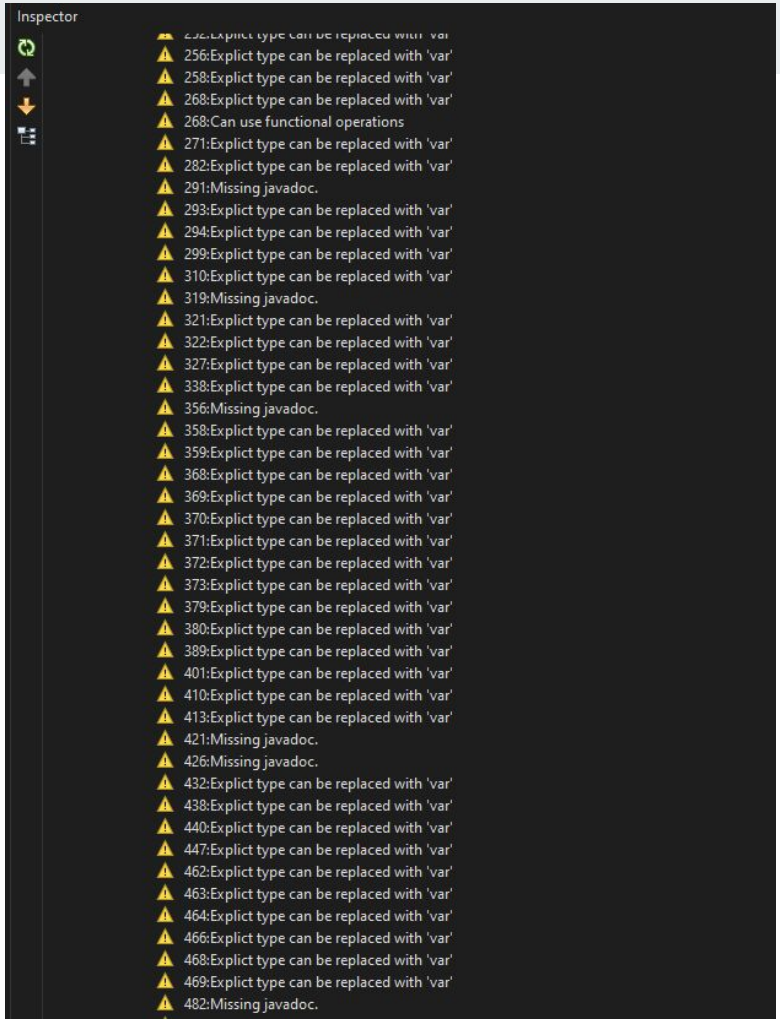
On my bot the issues that pop up after running the Analysis tool on my file are mostly Syntax suggestions and missing documentations on function.

The Static Analysis tool points out the syntax suggestions and upon clicking on the line number it will give several option in a dropdown menu on what I can change to make the code more precise.

Static Analysis

Matthew: Used Netbeans built in Static Analysis tool.

During development the built in static analysis frequently suggests various other ways of writing a line of code including warnings if something is not ideal. Due to this allowing me to resolve issues as they arise when running the static analysis tool separately it almost entirely finds small notes like changing explicit variable types to var or adding javadocs.



Plans Going Forward

Our plan is to further develop our bots to become more robust and optimized. As of now, our bots are only fairly effective maybe against particular bots or on specific maps. Our hope is to broaden the range of effectiveness of our bots. We also plan to play our bots against each other to continually develop and improve them.

Additional Items for Grading

Meeting Attendance Sheet

10/19/23 - All members attended

10/24/23 - All members attended except Ilias

Matthew Berry

Ohio ID, Github Username: mb135821, MatthewBerry135821

Contributions: Further individually implemented MINS_Bot including changes such as simulated decisions and performance improvements

https://github.com/OU-CS3560/microrts-f23/commit/252bc35ade311e829479dd20ec152974f7ef7aac

Akshay Patel

Ohio ID, Github Username: sp550519, Shayz614

Contributions: Created a new AI bot based on Our Initial MINS bot and mayari bot and did some bug fixing and refactoring, We are keeping our Ai's in different branches until we decide to move one that has most promise to the main branch.

https://github.com/OU-CS3560/microrts-f23/commit/c7f1249445c60243ef3797141a01328573ccf439

https://github.com/OU-CS3560/microrts-f23/commit/cbca101afb1f99bcd36508dd435e2de7514c03b9

Ilias Baktybek

Ohio ID: ib873519, Github Username: iliasbaktybek

Contributions:

Miguel Quemado

Ohio ID: mq003322, Github Username: MQUEMADO16

Contributions:

Created new bot based on main and utilizing methods for pathfinding found there. Implemented an attempt at a proxy light unit rush.

Commits:
https://github.com/OU-CS3560/microrts-f23/commits/miguel-branch