

## Forum:[Hack Forums](#)

Thread: [Using Stacks in Java](#)

**Failzorro** - March 8, 2018, 12:11 pm

I have created a program that takes inputs via the Eclipse console to control a physical Robot (Finch). The robot is able to move Forward/Backward/Left/Right as the user commands it. I am trying to implement a retrace feature (as shown in the "T" case) which would enable the user to input an 'n' integer value that would decide how many previous commands should be re-executed, starting from the last command. It should then execute the 'n' number of commands, one after the other. Only showing the Left movement part of the code as the rest of the code is a lot bulky.

Code case "L":

```
System.out.println("Left Selected, Please set duration of movement after Left turn");
myFinch.setWheelVelocities(-150, 150, 1000);
Scanner in8 = new Scanner(System.in);
int ldura = in8.nextInt();
if (ldura < 0 || ldura > 6) {
    System.out.println("Please enter an input greater than 0 that does not exceed 6");
} else
    System.out.println("Please set speed of finch after turn");
Scanner in9 = new Scanner(System.in);
int spdat2 = in9.nextInt();
System.out.println("Your command will now execute");
int ldur = ldura * 1000;
myFinch.setWheelVelocities(spdat2, spdat2, ldur);
int spdat20 = spdat2;
FinchRemoteControl lmove = new FinchRemoteControl(spdat2, spdat20, ldur);
st.push(lmove);
break;
case "T":
    System.out.println("Retrace Selected, Please set number of previous commands to retrace");
    Scanner retrace = new Scanner(System.in);
    int navhistory = retrace.nextInt();
    if (navhistory > countst()) {
        System.out.println("Not enough previous command have been executed.");
    } else
        System.out.println("Last " + navhistory + " command(s) will be executed.");
    break;
default:
    System.out.println("Please enter a valid command.");
    break;
}
System.out.println("Please enter your next command.");
move = sc.next(); // read user input
}

// End program with finch.quit()
myFinch.setWheelVelocities(0, 0);
myFinch.quit();
```

```

System.exit(0);
}
}

```

Another HF user has been advising me on how to go about this, and he said using a stack would be the best way to go. The Finch robot uses the following method `myFinch.setWheelVelocities(sp, spd, dur)`; which takes 3 integer values, left wheel/right wheel and duration. I created objects for each F/B/L/R command to store these values as seen here:

```

Code FinchRemoteControl lmove = new FinchRemoteControl(spdat2,spdat20, ldur);
Edit: Here's what i've come up with,
Code int numberOfPop = 0;
while (numberOfPop < navhistory) {
    st.pop();
    [color=##ff3333]//After it pops here, I would need to read what is popped and use the object values
from the popped item to form a
        "myFinch.setWheelVelocities(lmove);" method of some sort. The error i'm
getting is "The method setWheelVelocities(int, int) in the type Finch is not applicable for the
arguments (FinchRemoteControl)"[/color]
    numberOfPop++;
}
Any help would be appreciated, thanks.

```

**matt** - March 10, 2018, 7:49 am

You can just use an ArrayList and then iterate over the array starting at N'th element until the end of the array something like

```

Code ArrayList<String> commands = new ArrayList<String>();
String cmd = "turn left";
execute(cmd)
commands.add(cmd);

```

```

then to re-run the commands
for(int i = STARTING_INDEX; i < commands.size(); i++) {
    execute(commands.get(i));
}

```

However you might want to iterate over the list backwards because you want to execute the last commands first

**ellisplait** - March 12, 2018, 3:13 pm

You are pushing objects of type FinchRemoteControl:

```

Code FinchRemoteControl lmove = new FinchRemoteControl(spdat2,spdat20, ldur);
st.push(lmove);
The FinchRemoteControl does not have the method setWheelVelocities.

```

You seem to expect it to pop another class -- the class that has the `setWheelVelocities` method.

You need to post the full source or elaborate the issue if you need further help.

**Xoreaux** - March 13, 2018, 10:58 pm

(Mar 8, 2018 01:11 PM)Failzorro Wrote: [\[CITE\]](#) I have created a program that takes inputs via the

Eclipse console to control a physical Robot (Finch). The robot is able to move Forward/Backward/Left/Right as the user commands it. I am trying to implement a retrace feature (as shown in the "T" case) which would enable the user to input an 'n' integer value that would decide how many previous commands should be re-executed, starting from the last command. It should then execute the 'n' number of commands, one after the other. Only showing the Left movement part of the code as the rest of the code is a lot bulky.

Code case "L":

```
System.out.println("Left Selected, Please set duration of movement after Left turn");
myFinch.setWheelVelocities(-150, 150, 1000);
Scanner in8 = new Scanner(System.in);
int ldura = in8.nextInt();
if (ldura < 0 || ldura > 6) {
    System.out.println("Please enter an input greater than 0 that does not exceed 6");
} else
    System.out.println("Please set speed of finch after turn");
Scanner in9 = new Scanner(System.in);
int spdat2 = in9.nextInt();
System.out.println("Your command will now execute");
int ldur = ldura * 1000;
myFinch.setWheelVelocities(spdat2, spdat2, ldur);
int spdat20 = spdat2;
FinchRemoteControl lmove = new FinchRemoteControl(spdat2, spdat20, ldur);
st.push(lmove);
break;
case "T":
    System.out.println("Retrace Selected, Please set number of previous commands to retrace");
    Scanner retrace = new Scanner(System.in);
    int navhistory = retrace.nextInt();
    if (navhistory > countst()) {
        System.out.println("Not enough previous command have been executed.");
    } else
        System.out.println("Last " + navhistory + " command(s) will be executed.");
    break;
default:
    System.out.println("Please enter a valid command.");
    break;
}
System.out.println("Please enter your next command.");
move = sc.next(); // read user input
}

// End program with finch.quit()
myFinch.setWheelVelocities(0, 0);
myFinch.quit();
System.exit(0);
}
}
```

Another HF user has been advising me on how to go about this, and he said using a stack would be the best way to go. The Finch robot uses the following method `myFinch.setWheelVelocities(spdl, spd, dur)`; which takes 3 integer values, left wheel/right wheel and duration. I created objects for

each F/B/L/R command to store these values as seen here:

```
Code FinchRemoteControl lmove = new FinchRemoteControl(spdat2,spdat20, ldur);
Edit: Here's what i've come up with,
Code int numberOfPop = 0;
while (numberOfPop < navhistory) {
    st.pop();
    [color=##ff3333]//After it pops here, I would need to read what is popped and use the object values
from the popped item to form a
        "myFinch.setWheelVelocities(lmove);" method of some sort. The error i'm
getting is "The method setWheelVelocities(int, int) in the type Finch is not applicable for the
arguments (FinchRemoteControl)"[/color]
    numberOfPop++;
}
```

Any help would be appreciated, thanks.

You are trying to pass the object to the method but you need to be passing the integer values. You said you are using the object to store the three values so create accessors for those fields and then use them to access the values.

Instead of:

```
Code myFinch.setWheelVelocities(lmove);
It would look something like this:
```

```
Code myFinch.setWheelVelocities(lmove.getSpDat2(), lmove.getSpDat20, lmove.getLDur());
I'm going based off of the code that is visible. If you want a more comprehensive answer then you
will need to publish the full source.
```