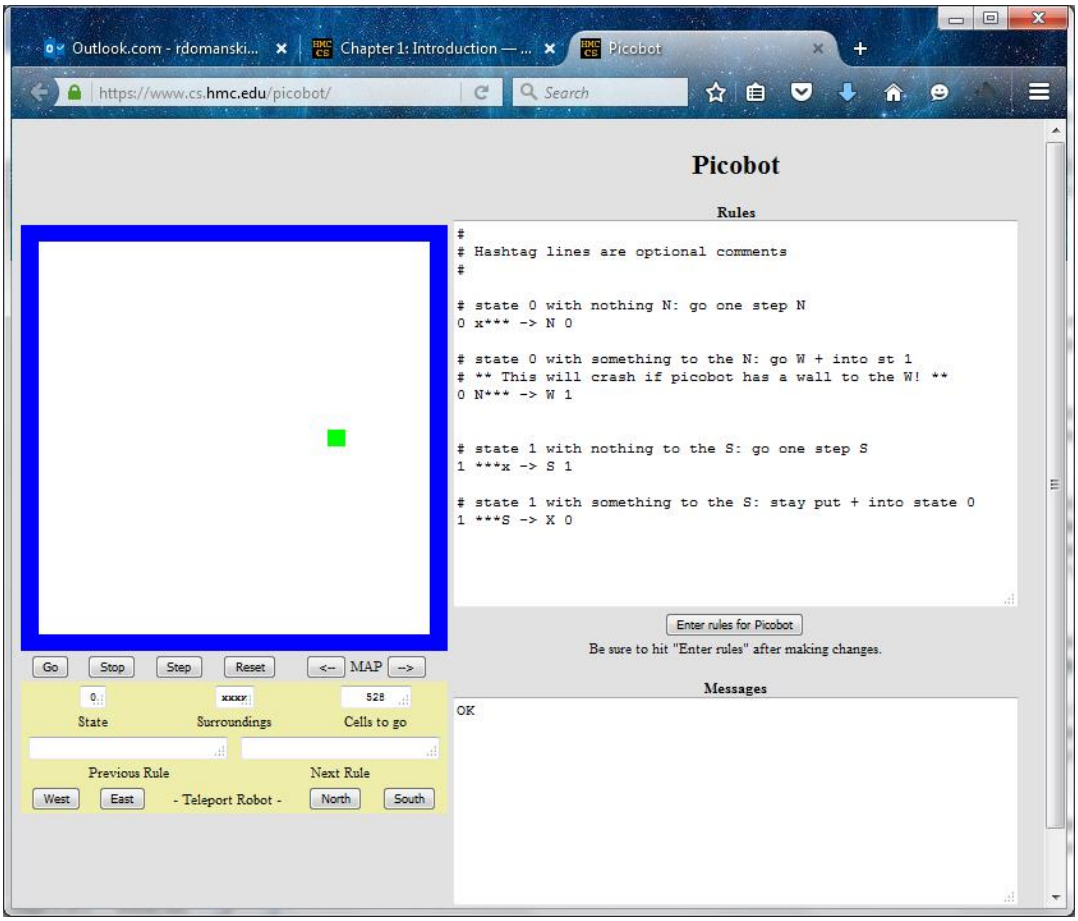
Lab Activity Just for fun – "State", "Abstraction", and Picobot

Let's tackle a serious computer science problem before we get sucked into a discussion of a full-blown programming language. This will be new and fun – whether or not you have programmed before. So head over to http://www.cs.hmc.edu/picobot.

Imagine yourself as a Roomba vacuum cleaner named Picobot: your goal is to vacuum all of the free space around you – without missing any part of the room. The robotics community calls this ***the coverage problem***.



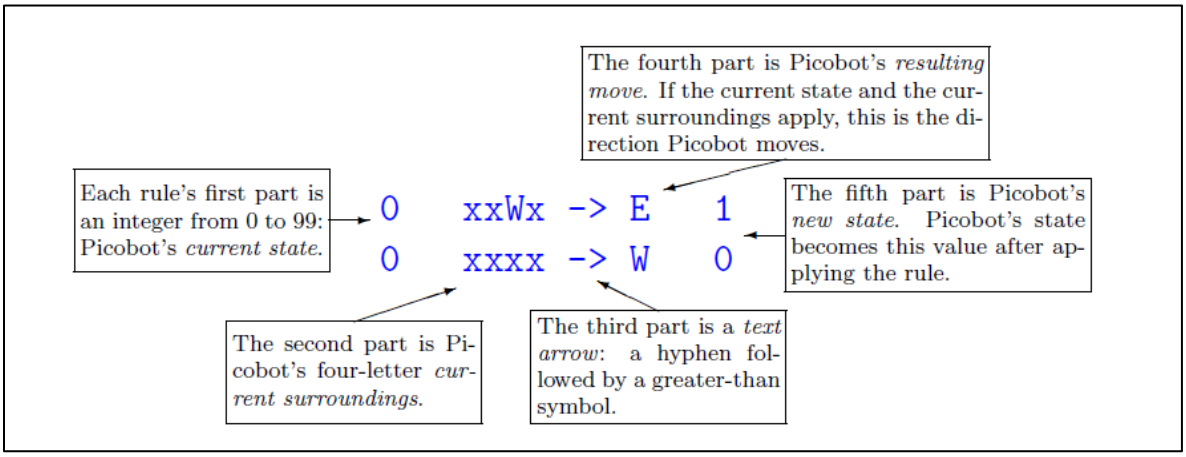
The way this all works is through an important concept called ***abstraction***. In this case, we are going to think of the environment as a grid made up of cells. You, as Picobot, occupy one grid square (the green one), and you can travel one step at a time in any of the four compass directions: North, East, West, or South. The surroundings are always reported as a string of four letters in **"NEWS"** order. If the cell to the north is empty, the letter in the first position is an `x`. If the cell to the north is occupied, the letter in that first position is an N, etc.

But how does Picobot "know" whether it is moving north or some other direction? Picobot doesn't have an innate sense of direction. Instead, we make use of a powerful concept called ***state***. The state of a computer (or a person or almost any other thing) is simply its current condition: on or off, happy or sad, underwater or in outer space, etc. In computer science, we often use "state" to refer to the internal information that describes what a computer is doing.

Picobot's state is extremely simple: it is a single number in the range 0-99. Somewhat surprisingly, that's enough to give Picobot some pretty complex behaviors. **Picobot always starts in state 0.**

Although Picobot's state is numeric, it's helpful to think of it in English terms. For example, we might think of state 0 as meaning "I'm heading north until I can't go any further." However, it's important to note that none of the state numbers has any special built-in meaning; it is up to us to make those decisions.

As far as giving Picobot specific instructions, Picobot moves by following a set of commands that specify actions and possible state changes. Which command Picobot follows depends on its current state and its current surroundings.



Here are two examples:

0 xxWx -> E 1

re-expressed in English, says "If I'm in state 0 and only my western neighbor contains an obstacle, take one step east and change into state 1."

That's the syntax for how to give Picobot instructions. The one other useful piece of syntax would be that you can use the wildcard character * to indicate that we don't care about the surroundings in the given position (N, E, W, or S). For example,

1 N*x* -> W 0

Says "If I'm in state 1 with a wall to the North and no obstacle to the West, move one step West and enter state 0". Because of the wildcards, we want this command to be executed regardless of whether there is also an obstacle to the East or South.

Your assignment:

Develop a set of rules that direct Picobot to cover the entirety of the empty room found at http://www.cs.hmc.edu/picobot. The set of rules – that is, your program – should work regardless of how big the room is and regardless of where Picobot initially begins.

When you are finished, copy and paste your Picobot code into a text file (with your names included) and submit it.

Extra Credit: On the Picobot webpage, click the → arrow next to "MAP". Write another program to cover all of the empty cells in the maze.