Files/Spreadsheet Processing

10/21/2009

Opening Discussion

- Imagine a program that you use frequently. Does that program use files? How would the functionality of the program change if it couldn't use files?
- What do files give our programs that we don't have without them?

Objectives

- We have a new topic and new goals. We are going to see how programming can let you do things with tabular data that you might have problems doing with standard applications.
- First, we need to learn how to read information from a file and put it into our program.
- After that we will work on processing the data.

Spreadsheet Scenario

- We have a new scenario that we are going to work with called spreadsheet.
- This scenario has a world that shows a grid.
- The grid can be filled with numbers that are drawn on the world.
- What we want to do today is fill in the readFile() method so that it can read in CSV text files.

Numbers with Decimals

- The numbers we are dealing with might have decimals.
- For that reason, we can't use the int type.
- Instead we will use the double type. Double stands for double-precision floating point. These aren't real numbers in the mathematical sense, they have limited precision.
- Our table uses the Double class which is a wrapper for double so it can return null.

Reading from File

- We are going to use the java.util.Scanner class for reading from files.
- Let's look at the API entry for that.
- To read from a file we will use the constructor that takes a java.io.File object.
- Let's look at the API entry for File.
- By default Greenfoot looks for files in the directory of the current scenario.

Strategy

- Here is what we want to do in readFile() to read a CSV file.
- Make a scanner.
- While the scanner has more lines
  - Read a line
  - Split the line on commas
  - For each of the values
    - Add it to the table

Exceptions

- When things go wrong in Java the code throws exceptions. Files can have lots of things go wrong.
- Use a try block when you want to try to do something that might throw an exception then catch that exception.
  - try {
    - Statements
  - } catch(ExceptionType e) {
    - Statements
  - }

The while Loop

- Not everything we repeat in code is well modeled by counting. Reading all the lines in a file is an example.
- The while loop allows you to repeat something as long as a condition is true.
- while(condition) {
  - Statements
- }

The split Method

- One way to break a string up into parts is with the split(String delim) method of the String class.
- You pass it the delimiter to split on and it gives you back a String[].
- We will use split on the line we read and then have a for loop go through the array.

Minute Essay

- Do you have any questions about the material we covered today?