CSE 335 Project 4, Spring 2015

Description

This project is an extension of Project 3. You may reuse all of your classes from project 3 to represent a company hierarchy. You may also use all visitor classes to print, count, and search within the company hierarchy. You can modify your previous classes in any way as long as they continue to follow the visitor and composite pattern.

Your goal is to build the company hierarchy from an input file given through command line arguments. You must write a parser to parse the input and use builder pattern to build the resulting hierarchy. The first line of the input will be the company name. The rest of the input file will be a comma-separated history database of employees and groups. The first number of each line represents whether we are adding something (1), removing something (2), or disbanding a group (3). If we are adding or removing something, the 2nd number determines what type of person: employee (1) or group (2). For employees, information is given in the order: last name, first name, job title, group. For groups, information is given in the order: group name, supervising group name. Removing an entity means to pull it and all its children from the hierarchy while disbanding a group means to remove the group but to assign its children to the group directly above it. Inputs will never disband the root.

After building the company hierarchy, you will output an ASCII animation of the evolution of the company hierarchy over time. The animation will be done on the command line in arctic.cse.msu.edu, so you should test your program on command line as well. At the end of the animation, the parser should clear the animation away and leave printed the final hierarchy.

Your program must be able to handle invalid inputs such as adding an employee to a group that does not exist, adding employees before there is any existing group at all, etc.

If you have not finished project 3, you must finish the project (or use the TA code) before you work on project 4.

An example input file is provided. An explanation of the proper output for this input is given below.

Code Requirements

1. Your solution must be handed in with the following naming convention: partner1lastname_partner1firstname_partner2lastname_partner2firstname_cse335proj3.zip and will consist of a zip file of all deliverables.
2. Your program must free any memory that it allocates.
3. Your solution must include a separate .h and .cpp file for each class you implement.
4. You must implement a default constructor, a copy constructor, an assignment operator, and a destructor for each class.
5. Your program should compile and run on arctic.cse.msu.edu. You can write your own Makefile or you can create the project in Netbeans.
6. All data members should be private.

Deliverables

1. UML diagram in pdf or jpg format. UML must comply with your program.
2. Readme.txt including
   a. netID and full name of each group member
   b. individual specific contributions of each team member
   c. expected functionality of the project. If everything is working, simply say so. Elsewise, describe what is and is not working.
3. Your implementation

Handin Procedure

This project is due via “handin” (http://secure.cse.msu.edu/handin/) by 11:59 PM on 4/14/2015.

Example Output

The complete output for the sample input is given below.

Each time you make a change to the existing company, you will print out the hierarchy again. You should animate it so that every 3-5 seconds, the result of reading in a new line of the input and processing it appears. For better visualization, because we are unable to show animations on printed paper, we have included ----- everywhere the screen would be cleared. The very last repeated output comes from the Parser when it has reached the end of the file and is done parsing.

```
+ Team Apple

--------------------------------

+ Team Apple

  CEO: Tim Cook

--------------------------------

+ Team Apple

  CEO: Tim Cook

  + Team VP

--------------------------------

+ Team Apple

  CEO: Tim Cook
  + Team VP
    VP of Engineering: Craig Federighi
---------------------------------
+ Team Apple
  CEO: Tim Cook
  + Team VP
    VP of Engineering: Craig Federighi
    VP of Marketing: Phillip Schiller
---------------------------------
+ Team Apple
  CEO: Tim Cook
  + Team VP
    VP of Engineering: Craig Federighi
    VP of Marketing: Phillip Schiller
    + Team Smartwatch
---------------------------------
+ Team Apple
  CEO: Tim Cook
  + Team VP
    VP of Engineering: Craig Federighi
    VP of Marketing: Phillip Schiller
    + Team Smartwatch
    + Team iPad
---------------------------------
+ Team Apple
```

```
  CEO: Tim Cook
  + Team VP
    VP of Engineering: Craig Federighi
    VP of Marketing: Phillip Schiller
    + Team Smartwatch
    + Team iPad
      iPad Designer: Bob Smith
---------------------------------
ERROR: parent group Chefs does not exist. Omitting this entry.
+ Team Apple
  CEO: Tim Cook
  + Team VP
    VP of Engineering: Craig Federighi
    VP of Marketing: Phillip Schiller
    + Team Smartwatch
    + Team iPad
      iPad Designer: Bob Smith
---------------------------------
+ Team Apple
  CEO: Tim Cook
  + Team VP
    VP of Engineering: Craig Federighi
    VP of Marketing: Phillip Schiller
    + Team Smartwatch
---------------------------------
```

```
+ Team Apple
  CEO: Tim Cook
  VP of Engineering: Craig Federighi
  VP of Marketing: Phillip Schiller
  + Team Smartwatch
---------------------------------
+ Team Apple
  CEO: Tim Cook
  VP of Engineering: Craig Federighi
  VP of Marketing: Phillip Schiller
  + Team Smartwatch
```