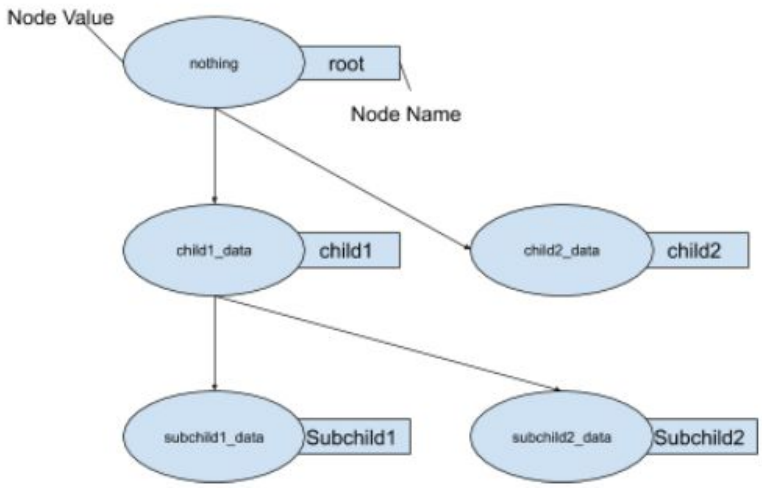
Instructions:

1. Use your favorite editor and programming language [preferably Java or C#] to implement the below questions.

2. Create a project with your name and write executable code in it. Zip the same and send it on email.

Question:

Hierarchical data store: This is a store where the data has a hierarchy along with data for example you can create a node called services with some String data associated to it which can in turn contain further nodes. See the diagram below.



String path example: “/root/child1/subchild1” or “/root/child2”

You can do following operations on a hierarchical data store

1. Create a node: Create a node with a path and data - create(String path, String data)

2. Update a node: Update a value of a node - update(path, data)

3. Delete a node: Delete a node - delete(path)

4. Get: Get data from a node - data get(path)

5. List: List all direct child nodes for a given node - nodes[] list(path)

Apart from this anyone can choose to listen to a particular node and get an event whenever something changes. Listener should get events (creation/deletion/updation) of all direct and indirect child nodes of path along with any changes in data of the node itself. - addListner(path, listener)

Design interfaces in Java/C# to define contracts for interaction with the system as well as Listeners & events. Implement the hierarchical data store. The data can reside in memory and are only be persisted during the lifecycle of the program. Write a main method to test following scenarios

1. Add a node - root with a string data “nothing”

2. Attach a listener which prints all events to root. [Bonus]

3. Add two child nodes to root - child1 with a string data “childdata 1” & child2 with a string data “childdata 2”

4. Add one child node to child1 - subchild1 with a string data “subchild1”.

5. Get and print the data for all the nodes.

6. List all the child nodes for root.

7. Delete the node child2.

8. Keep into consideration that this system can be used in a multi-threaded environment [Bonus]