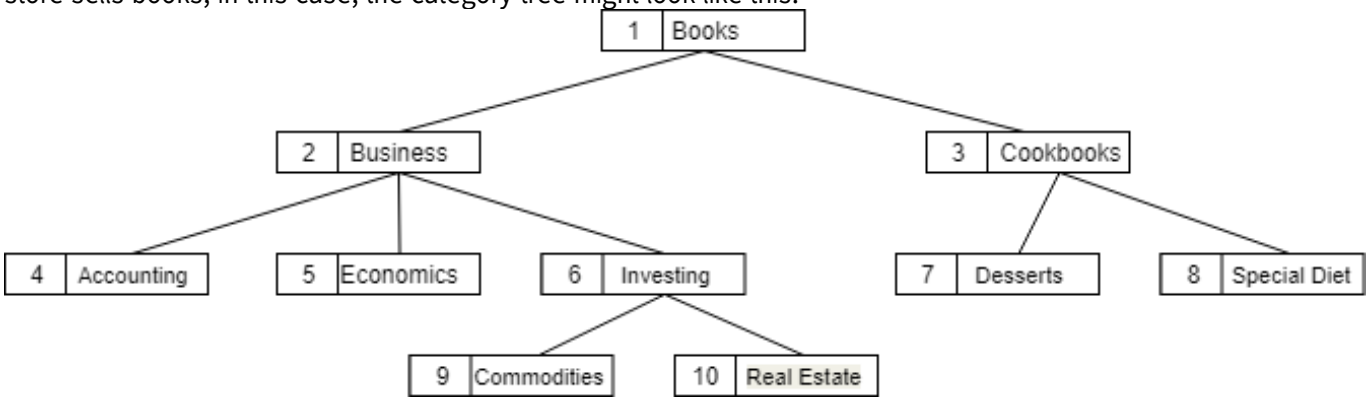
Article
Nikolay Solovyev · Aug 1, 2019 3m read

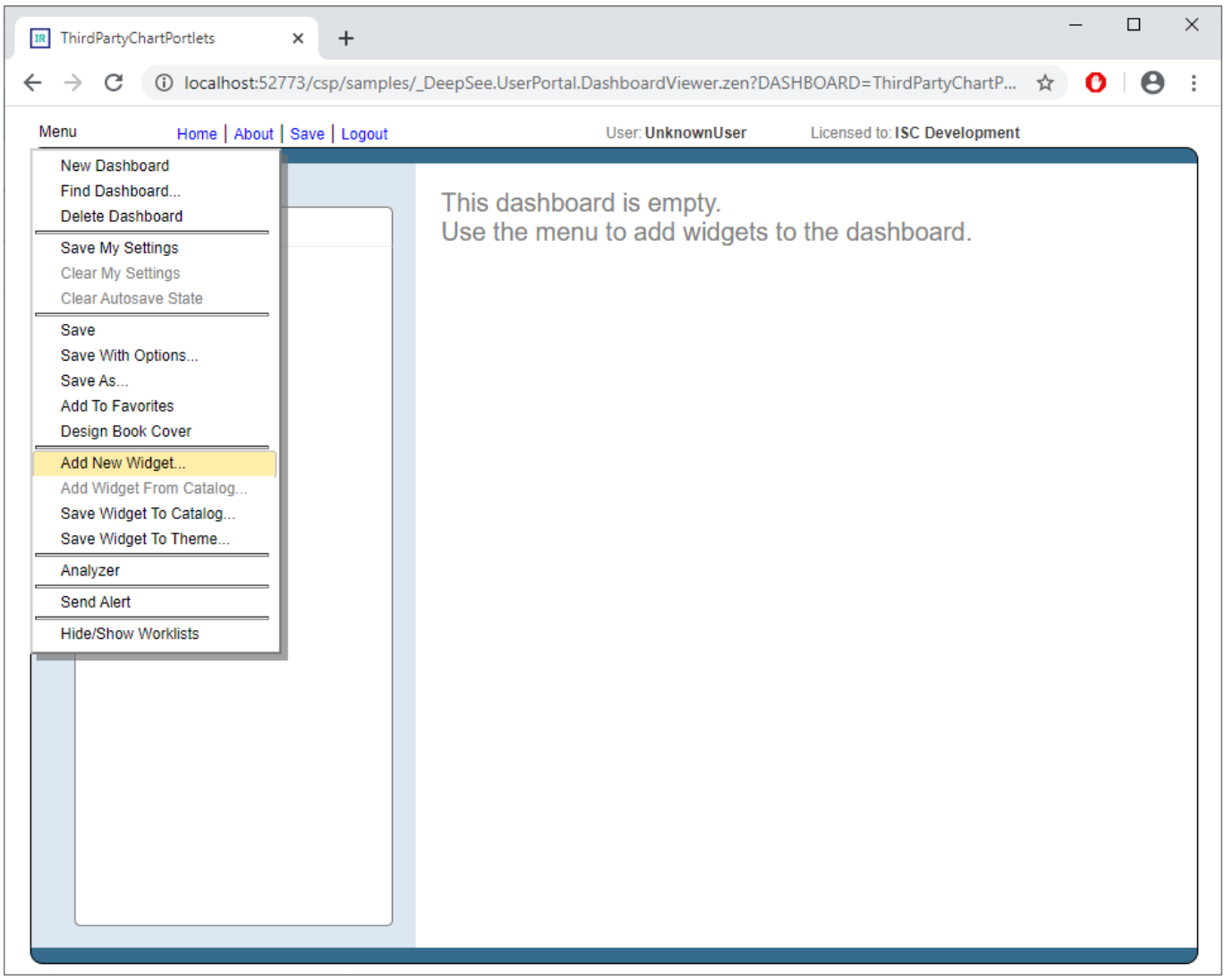
Open Exchange

Nested set model for ObjectScript

In many projects I was faced with storing hierarchical data (tree) in classes.
By tree, I mean such data, where each node has a parent node — an object of the same class.
Many examples of such data can be given. For example, a catalog in the online store. Suppose that this online store sells books, in this case, the category tree might look like this:



The number in front of the name is the category ID.
For storage, you can create classes:

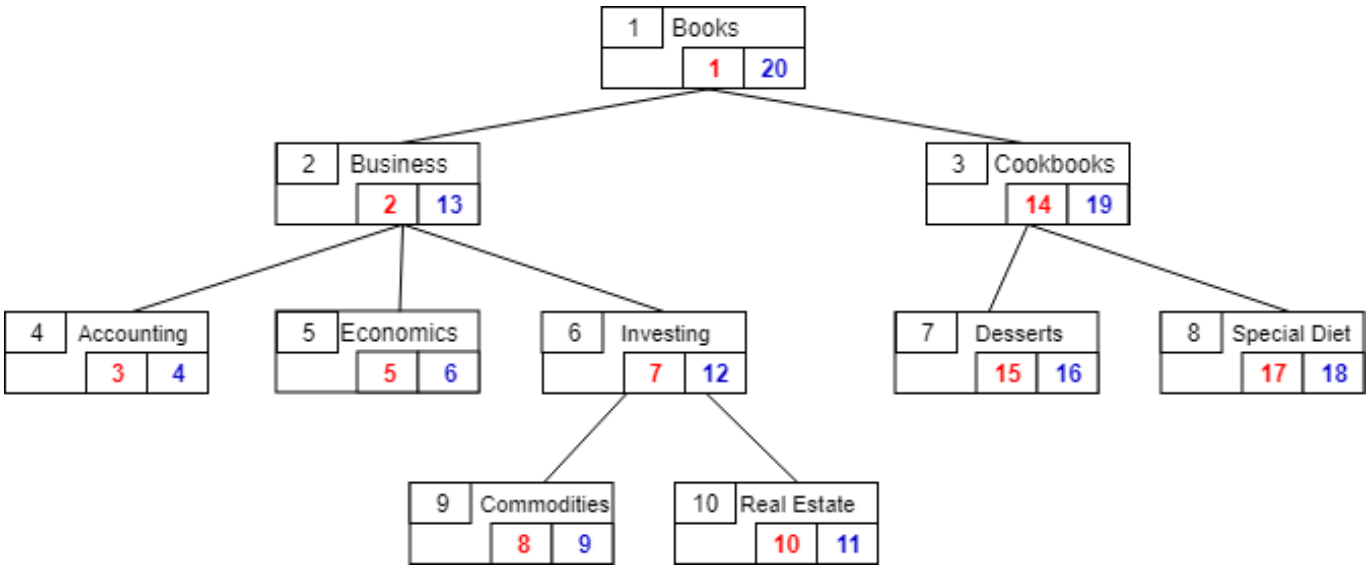


The MyApp.Category class is used to store the category tree and contains the property “ Parent ” - reference to the same class.

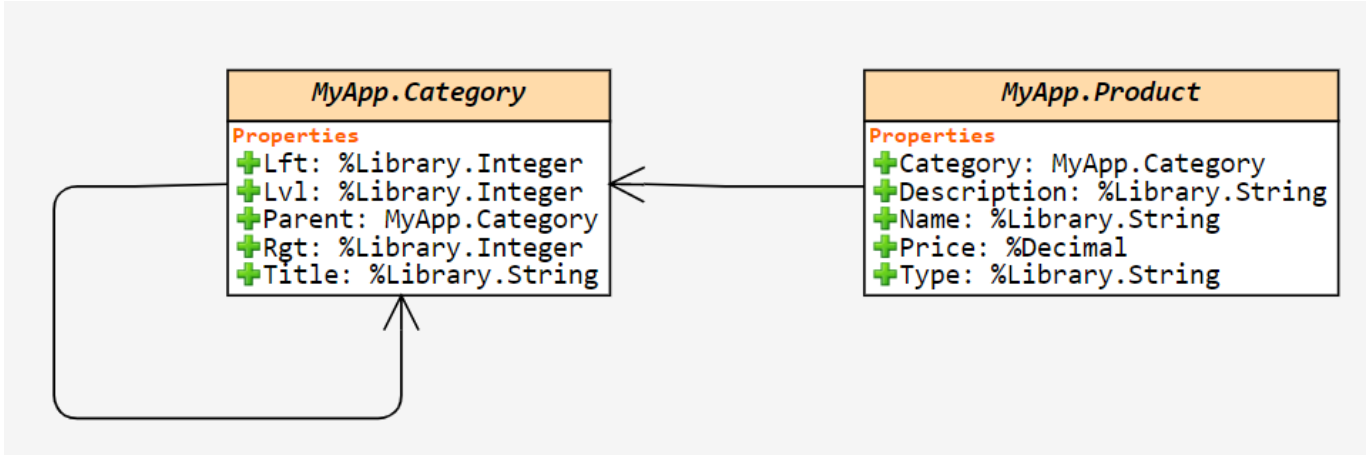
But it ’ s not convenient to work with such implementations. For example, if you need to find all the products that belong to a certain category or any of its subcategories. For example, all books from the category Business or its subcategories.
Or you need to get the categories path for one of the products.

There are several models that allow you to add just a little extra information to the stored classes to make your life much easier. The nested set model is among them.

This model adds 2 additional fields to each node: the left and right keys.



Left (Lft) is marked in red and RIGHT (Rgt) in blue.
These numbers are set according to a traversal, which visits each node twice.
Usually, one more field is added - level (Lvl).
I use abbreviated (Lvl, Lft) names instead of full names(LEVEL, LEFT) to avoid conflicts with SQL reserved words.



Now to get all the categories that are children of the category “ Business ” , this can be done with a simple SQL query.

```
&sql(SELECT * FROM MyApp.Category WHERE Lft>=:left AND Rgt<=:right)
```

specifying left = 2, right = 13 - the value for the Lft and Rgt fields of the category “ Business ” .

Since using this model it is very easy to extract data, but when adding / deleting / modifying, it is necessary to adjust the values of additional fields (and this requires additional costs), this model is best suited when the number of reads (data retrieval) exceeds the number of data modification operations.

There are implementations of the nested set model for different languages (frameworks), now there is an **implementation for objectscript**.
Github repository: https://github.com/nsolov/iris-nestedset

This implementation allows:

- automatically sets the values of the additional fields Level, Right, Left when:
    - adding new data (no matter how you add data via SQL or saving an object using the %Save method)
    - deleting a node (deleting a node deletes all its sub-nodes)
- reorder nodes at one level of the tree
- supports work with several trees (with several roots). Even if you are working with only one tree, you must define the Root property, which will contain the numeric identifier of the root node.

Restrictions:

- only numeric identifiers are supported
- it is not allowed to specify SqlFieldName for special fields (Parent, Rgt, Lvl, Lft, Root).

How to Install

1. Import classes from CDEV package from the repository.
2. Create a persistent class (for example, MyApp.Category) and add CDEV.NestedSet as a superclass to your class.
3. In your class (MyApp.Category) add properties: Parent, Rgt, Lvl, Lft, Root.
4. Override parameters PARENT, LEVEL, ROOT, LEFT, RIGHT. As the values of these parameters, you must specify the names of the corresponding properties in your class.
5. Add an index on Root, Lft, Rgt

```
Index TreeIndex On (Root, Lft, Rgt);
```

6. Compile your class

How to use

1. Use %New(), %Save(), %DeleteId() or SQL queries as usual.
   To add new node specify Parent property, if you leave Parent property blank - new root will be created

2. To add new node you can also use AddFirstChild() or AddLastChild()

3. Use MoveUp() or MoveDown() to reorder siblings

Take a look at the MyApp.Category class - this is an example of catalog implementation. In this example, the PrintTree () method displays the subtree, and PrintPath () displays the path to the node in the terminal.

```
NS > do ##class(MyApp.Category).PrintTree(1)

 1 Books(left=1, right=20)
 * 2 Business(left=2, right=13)
 * * 4 Accounting(left=3, right=4)
 * * 6 Investing(left=5, right=10)
 * * * 9 Commodities(left=6, right=7)
 * * * 10 Real Estate(left=8, right=9)
 * * 5 Economics(left=11, right=12)
 * 3 Cookbooks(left=14, right=19)
 * * 7 Dessetrs(left=15, right=16)
 * * 8 Special Diet(left=17, right=18)

NS > do ##class(MyApp.Category).PrintTree(6)

 6 Investing(left=5, right=10)
 * 9 Commodities(left=6, right=7)
 * 10 Real Estate(left=8, right=9)

NS > do ##class(MyApp.Category).PrintPath(10)

Books > Business > Investing > Real Estate
```

If you use other models for tree data, please write in the comments!

#Beginner #Code Snippet #Data Model #Databases #ObjectScript #InterSystems IRIS
Check the related application on InterSystems Open Exchange