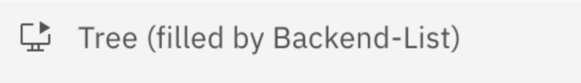
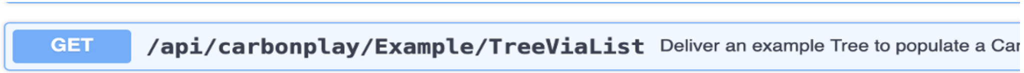
IBM Carbon TreeView

Example 03

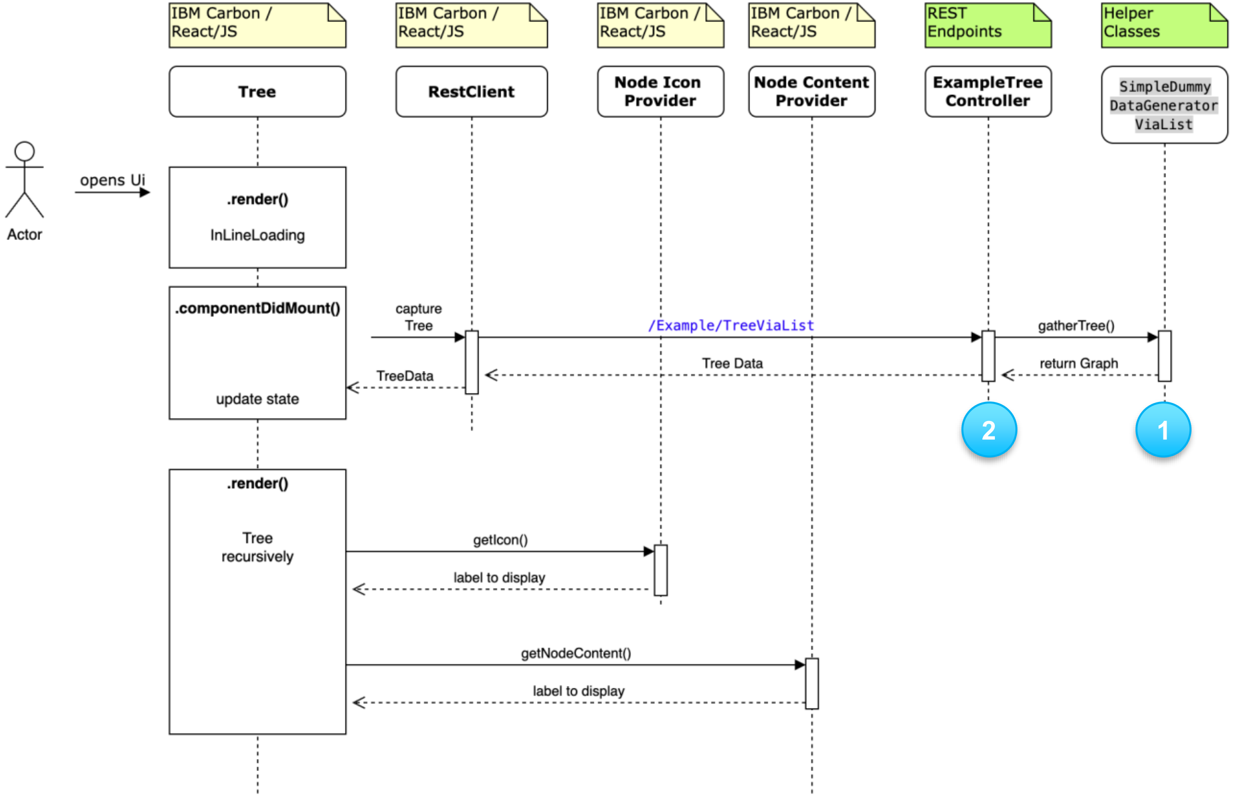


Tree (filled by Backend-List)



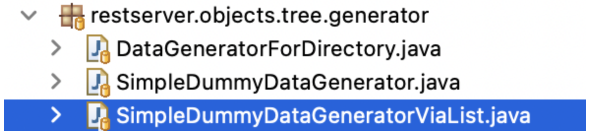
GET /api/carbonplay/Example/TreeViaList Deliver an example Tree to populate a Car

Sequence Diagram



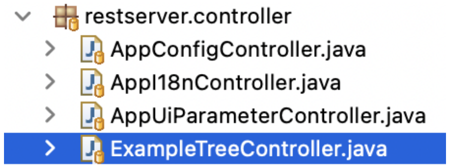
Server-sided Sources

1



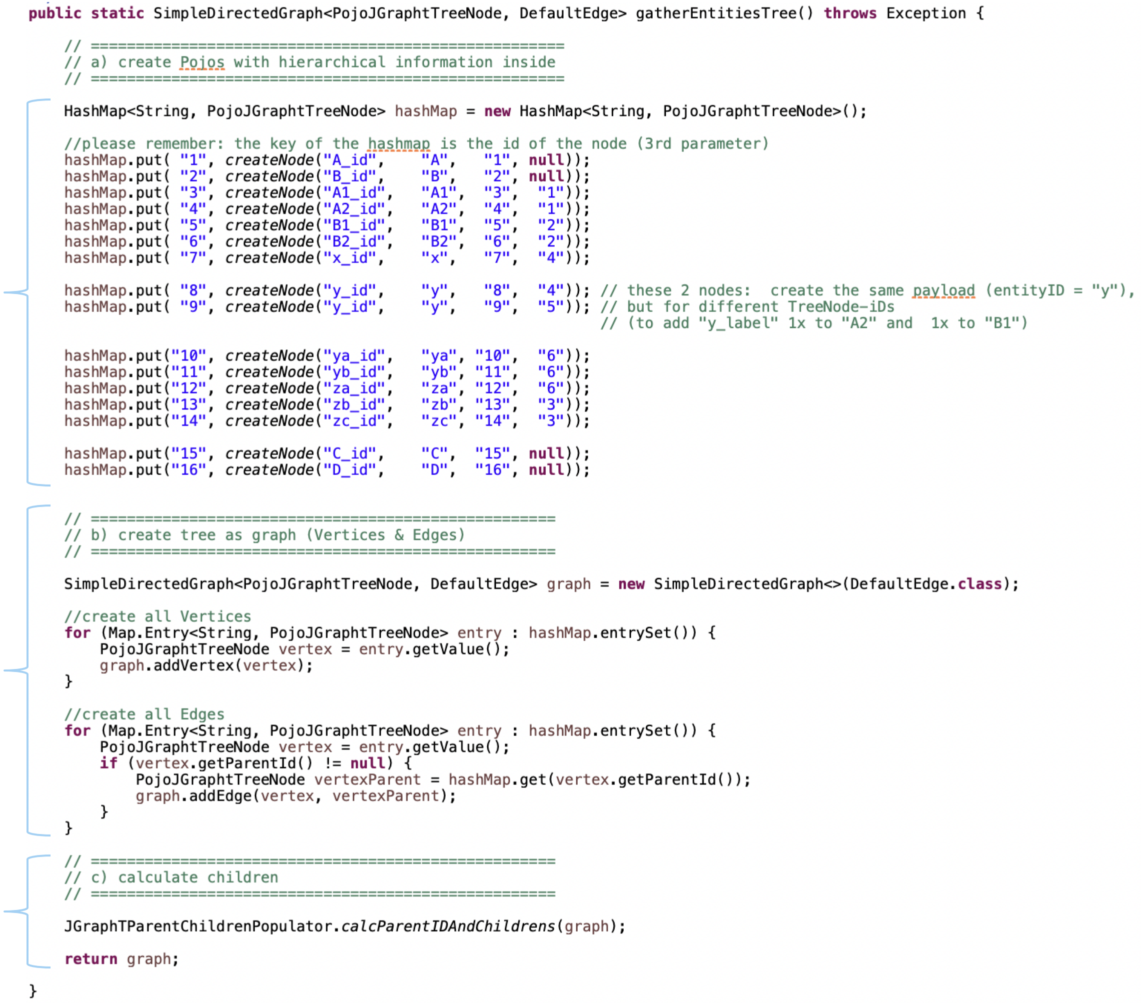
```
restserver.objects.tree.generator
    DataGeneratorForDirectory.java
    SimpleDummyDataGenerator.java
    SimpleDummyDataGeneratorViaList.java
```

2



```
restserver.controller
    AppConfigController.java
    AppI18nController.java
    AppUiParameterController.java
    ExampleTreeController.java
```

1



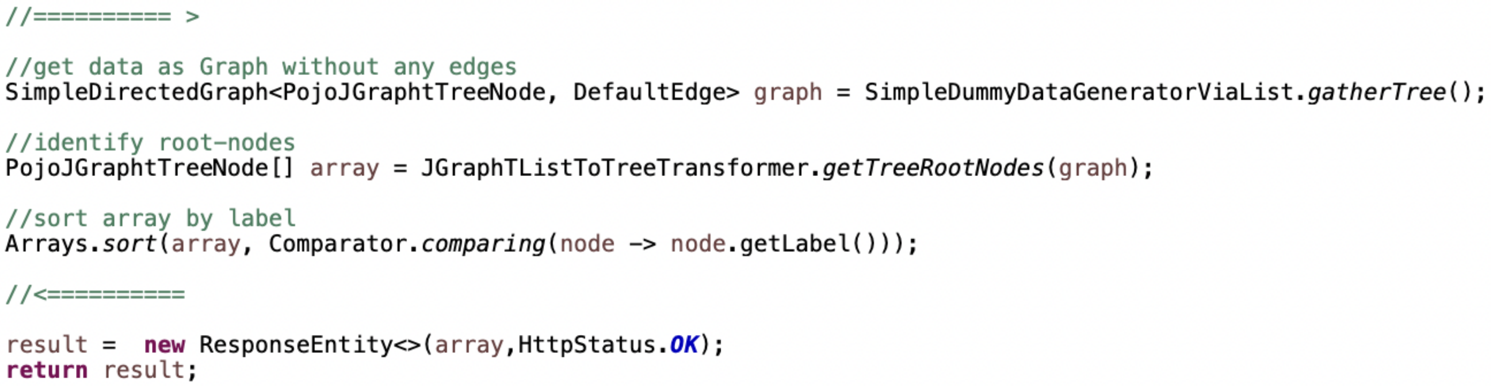
```java
public static SimpleDirectedGraph<PojoJGraphtTreeNode, DefaultEdge> gatherEntitiesTree() throws Exception {

    // ======================================================
    // a) create Pojos with hierarchical information inside
    // ======================================================

    HashMap<String, PojoJGraphtTreeNode> hashMap = new HashMap<String, PojoJGraphtTreeNode>();

    //please remember: the key of the hashmap is the id of the node (3rd parameter)
    hashMap.put( "1", createNode("A_id",    "A",   "1", null));
    hashMap.put( "2", createNode("B_id",    "B",   "2", null));
    hashMap.put( "3", createNode("A1_id",   "A1",  "3",  "1"));
    hashMap.put( "4", createNode("A2_id",   "A2",  "4",  "1"));
    hashMap.put( "5", createNode("B1_id",   "B1",  "5",  "2"));
    hashMap.put( "6", createNode("B2_id",   "B2",  "6",  "2"));
    hashMap.put( "7", createNode("x_id",    "x",   "7",  "4"));

    hashMap.put( "8", createNode("y_id",    "y",   "8",  "4")); // these 2 nodes:  create the same payload (entityID = "y"),
    hashMap.put( "9", createNode("y_id",    "y",   "9",  "5")); // but for different TreeNode-iDs
                                                                // (to add "y_label" 1x to "A2" and  1x to "B1")

    hashMap.put("10", createNode("ya_id",   "ya", "10",  "6"));
    hashMap.put("11", createNode("yb_id",   "yb", "11",  "6"));
    hashMap.put("12", createNode("za_id",   "za", "12",  "6"));
    hashMap.put("13", createNode("zb_id",   "zb", "13",  "3"));
    hashMap.put("14", createNode("zc_id",   "zc", "14",  "3"));

    hashMap.put("15", createNode("C_id",    "C",  "15", null));
    hashMap.put("16", createNode("D_id",    "D",  "16", null));

    // =====================================================
    // b) create tree as graph (Vertices & Edges)
    // =====================================================

    SimpleDirectedGraph<PojoJGraphtTreeNode, DefaultEdge> graph = new SimpleDirectedGraph<>(DefaultEdge.class);

    //create all Vertices
    for (Map.Entry<String, PojoJGraphtTreeNode> entry : hashMap.entrySet()) {
        PojoJGraphtTreeNode vertex = entry.getValue();
        graph.addVertex(vertex);
    }

    //create all Edges
    for (Map.Entry<String, PojoJGraphtTreeNode> entry : hashMap.entrySet()) {
        PojoJGraphtTreeNode vertex = entry.getValue();
        if (vertex.getParentId() != null) {
            PojoJGraphtTreeNode vertexParent = hashMap.get(vertex.getParentId());
            graph.addEdge(vertex, vertexParent);
        }
    }

    // =====================================================
    // c) calculate children
    // =====================================================

    JGraphTParentChildrenPopulator.calcParentIDAndChildrens(graph);

    return graph;

}
```

This might come from a database

Build Graph via JGraphT

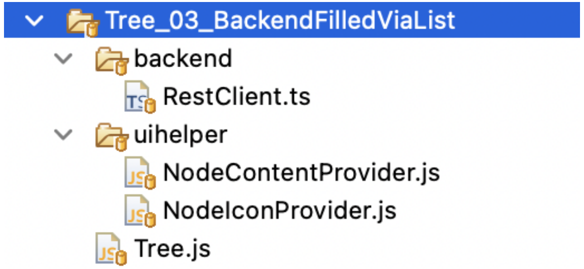
Grant ParentID & Children

2



```
//========== >

//get data as Graph without any edges
SimpleDirectedGraph<PojoJGraphtTreeNode, DefaultEdge> graph = SimpleDummyDataGeneratorViaList.gatherTree();

//identify root-nodes
PojoJGraphtTreeNode[] array = JGraphTListToTreeTransformer.getTreeRootNodes(graph);

//sort array by label
Arrays.sort(array, Comparator.comparing(node -> node.getLabel()));

//<==========

result =  new ResponseEntity<>(array,HttpStatus.OK);
return result;
```

Client-sided Sources



```
Tree_03_BackendFilledViaList
    backend
        RestClient.ts
    uihelper
        NodeContentProvider.js
        NodeIconProvider.js
    Tree.js
```