链滴

jsonArray 转 tree

```
/**
 * * @param data 入口参数data,平行数组
 * @param key id字段
 * @param parentKey 父字段
 * @param map 需要将原始属性名称转换为什么名称 {"text":"title","id":"id0"};
 */export default function (data, key, parentKey, map) {
 this.data = data;
 this.key = key;
 this.parentKey = parentKey;
 this.treeParentKey = parentKey; //parentKey要转换成什么属性名称
 this.treeKey = key; //key要转换成什么属性名称
 this.map = map;
 if (map) {
 if (map[key]) this.treeKey = map[key];
 }
 /**
 * 转成tree
 * @returns {Array}
 */ this.toTree = function () {
 let data = this.data;
 let pos = {};
 let tree = [];
 let i = 0;
 while (data.length !== 0) {
 if (data[i][this.parentKey] === 0) {
 let _temp = this.copy(data[i]);
 tree.push(_temp);
 pos[data[i][this.key]] = [tree.length - 1];
 data.splice(i, 1);
 i--;
 } else {
 let posArr = pos[data[i][this.parentKey]];
 if (posArr !== undefined) {
 let obj = tree[posArr[0]];
 for (let j = 1; j < posArr.length; j++) {
 obj = obj.children[posArr[j]];
 }
 let _temp = this.copy(data[i]);
 obj.children.push(_temp);
 pos[data[i][this.key]] = posArr.concat([obj.children.length - 1]);
 data.splice(i, 1);
 i--;
 }
 } i++;
 if (i > data.length - 1) {
 i = 0;
 }
 } return tree;
 };
 /**
 * 拷贝
 * @param item
 * @returns {{children: Array}}
 */ this.copy = function (item) {
```

```
    let _temp = {
    children: []
};
    _temp[this.treeKey] = item[this.key];
    for (let _index in item) {
    if (_index !== this.key && _index !== this.parentKey) {
    let _property = item[_index];
    if ((!!this.map) && this.map[_index])
    _temp[this.map[_index]] = _property;
    else
    _temp[_index] = _property;
    }
}  return _temp;
    }
}
```