















Introduction to Reactlet Treeview

Yue Liu

<https://www.linkedin.com/in/leapon>

- A simple treeview
- Built using ReactJS
- Interface design
- Links

- ▼  website
 - ▼  images
 -  logo.png
 -  background.png
 -  index.html
 -  about.html
 -  product.html

- ▼  website
 - ▼ ☒  images
 - ☒  logo.png
 - ☒  background.png
 - ☒  index.html
 - ☐  about.html
 - ☐  product.html

React Classes:

Treeview

TreeNode



TreeviewCell



TreeviewTextBox



Structure

Treeview = TreeviewNode (+)

TreeviewNode = TreeviewCell (+) & TreeviewTextBox

Rectlet Treeview

▼ website

▼ images

div.treeview-node-container 1140px × 20px

 background.png index.html about.html product.html Elements Network Sources Timeline Profiles Resources Audits Console

```
▼ <div class="body-container container">
  ::before
  ▶ <div class="row">...</div>
  ▼ <div class="row">
    ::before
    ▼ <div class="col-md-12">
      ▼ <div id="treeview1">
        ▼ <div class="treeview-container" data-reactid=".0">
          ▼ <div data-reactid=".0.0">
            ▶ <div class="treeview-node-container" data-id="1e98dbb4a7c25dcd8a3d2e0403c02929" data-reactid=".0.0.$streenode-1e98dbb4a7c25dcd8a3d2e0403c02929-n-e-n">
              ...</div>
            ▼ <div class="treeview-node-container" data-id="e6a68aa636c86015161828bfc226db89" data-reactid=".0.0.$streenode-e6a68aa636c86015161828bfc226db89-n-e-n">
              ▶ <div class="treeview-cell-container" data-reactid=".0.0.$streenode-e6a68aa636c86015161828bfc226db89-n-e-n.$treeview-text-cell-space-1">_</div>
              ▶ <div class="treeview-cell-container" data-reactid=".0.0.$streenode-e6a68aa636c86015161828bfc226db89-n-e-n.$treeview-expand-cell">_</div>
              ▼ <div class="treeview-cell-container" data-reactid=".0.0.$streenode-e6a68aa636c86015161828bfc226db89-n-e-n.$treeview-icon-cell">
                ▼ <div class="treeview-cell-icon-container" data-reactid=".0.0.$streenode-e6a68aa636c86015161828bfc226db89-n-e-n.$treeview-icon-cell.0">
                  ▶ <i class="fa fa-folder-o" data-uid="e6a68aa636c86015161828bfc226db89-icon" data-reactid=".0.0.$streenode-e6a68aa636c86015161828bfc226db89-n-e-n.$treeview-icon-cell.0.0">_</i>
                  </div>
                </div>
              </div>
              ▼ <div class="treeview-textbox-container" data-reactid=".0.0.$streenode-e6a68aa636c86015161828bfc226db89-n-e-n.$treeview-textbox">
                <div class="treeview-textbox-content" data-uid="e6a68aa636c86015161828bfc226db89-textbox" data-reactid=".0.0.$streenode-e6a68aa636c86015161828bfc226db89-n-e-n.$treeview-textbox.0">images</div>
              </div>
            </div>
            ▶ <div class="treeview-node-container" data-id="0047c4e80bc395f6b6f06461da025479" data-reactid=".0.0.$streenode-0047c4e80bc395f6b6f06461da025479-n-n-n">
              ...</div>
```

Treeview Data

```
app.tree1Data = {
  expandLevel: 3,
  treedata: [
    {
      "name": "website",
      "iconClass": "fa fa-folder-o",
      "children": [
        {
          "name": "images",
          "iconClass": "fa fa-folder-o",
          "children": [
            {
              "name": "logo.png",
              "iconClass": "fa fa-file-image-o"
            },
            {
              "name": "background.png",
              "iconClass": "fa fa-file-image-o"
            }
          ]
        }
      ]
    },
    {
      "name": "index.html",
      "iconClass": "fa fa-file-text-o"
    },
    {
      "name": "about.html",
      "iconClass": "fa fa-file-text-o"
    },
    {
      "name": "product.html",
      "iconClass": "fa fa-file-text-o"
    }
  ]
};
```


Create treeview

```
app.treeview1 = React.renderComponent(  
  React.createElement(Treeview, {data: app.tree1Data}),  
  document.getElementById('treeview1')  
);
```

Data Input is validated via mixin

```
var Treeview = React.createClass({  
  name: 'treeview',  
  mixins: [getCommonMixin],  
  ...
```

Data input has data model

```
// attribute definitions
getAttributes: function() {
  var attributes = [
    { name:'boxClass', type:'string', required:false, defaultValue:"", note:'container CSS class' },
    { name:'dispatcher', type:'object', required:false, defaultValue:null, note:'flux dispatcher' },
    { name:'selectType', type:'string', required:false, defaultValue:'n', note:'select type:none/checkbox/radio' },
    { name:'expandLevel', type:'number', required:false, defaultValue:1, note:'levels to expand' },
    { name:'treedata', type:'array', required:false, defaultValue:[], note:'input treeview data' },
    // internal, don't pass in
    { name:'activeNode', type:'object', required:false, defaultValue:[], note:'the active treeview node' },
    { name:'treedataObject', type:'object', required:false, defaultValue:[], note:'treeview data object' },
    { name:'treedataCol', type:'array', required:false, defaultValue:[], note:'treeview data in hash array' }
  ];
  return attributes;
},
```

Mixin function uses data model to populate this.state

All React Classes follow this data input convention

- Treeview
- TreeviewNode
- TreeviewCell
- TreeviewTextBox

Reactlet components use same component design and API for ease of development.

Code and UI demo

Treeview is on NPM

<https://www.npmjs.com/package/reactlet-treeview>

More Reactlet components at:

<http://reactlet.com/>

Source code on github

<https://github.com/reactlet/treeview>

<https://github.com/reactlet/table>

<https://github.com/reactlet/calendar>

Questions ?