

AXTree

new AXTree()

[AXTree.js, line 4](#)

```
var myTree = new AXTree();
myTree.setConfig(classConfig:JSObject);
```

Version:

- v1.58

Author:

- tom@axisj.com

Extends

- [AXI](#)

Methods

static
[AXTree.js, line 4178](#)

AXTree.appendTree(itemIndex, item, subTree)

원하는 아이템 하위에 아이템을 추가합니다.

| Name | Type | Description |
|-----------|----------|-------------|
| itemIndex | Number | 부모아이템 인덱스 |
| item | JSObject | 부모아이템 |
| subTree | JSObject | 추가하려는 아이템 |

Example

```
1. // 선택아이템의 자식 추가하기
2. var obj = myTree.getSelectedList();
3. myTree.appendTree(obj.index, obj.item,
  [{nodeID:"N", nodenm frm.nodeName value,
  writer:"mondo", type:"file",
  parentcd:obj.item.nodeID}]);
4.
5. // 선택아이템의 형제 추가하기
6. var obj = myTree.getSelectedListParent();
7. myTree.appendTree(obj.index, obj.item,
  [{nodeID:"N", nodenm frm.nodeName value,
  writer:"mondo", type:"file", parentcd:
  (obj.item.nodeID|0)}]);
```

static

AXTree.clearFocus()

[AXTree.js, line 3693](#)

선택된 상태를 해제합니다.

Example

```
1. myTree.clearFocus();
```

static **AXTree.click**(itemIndex,
open, doNotCallBack) **{JSObject}**

 [AXTree.js, line 3780](#)

아이템인덱스의 아이템 선택, 확장, 클릭이벤트 발생 처리를 합니다.

| Name | Type | Description |
|---------------|---------|---|
| itemIndex | Number | index of Array |
| open | String | "open"이면 아이템개체 확장 |
| doNotCallBack | Boolean | <div>optional</div> 아이템 개체 확장 처리 후 클릭이벤트 발생 방지 |

Returns:

ID } 대상아이디가 오브젝트로 옵니다.

Example

```
1. var findIndex = null;
2. $.each(List, function(jindex, J){
3.     if(this.id == "findid"){
4.         findIndex = jindex;
5.         return false;
6.     }
7. });
8. if(findIndex != null){
9.     var focusItem = myTree.click(findIndex,
10. "open", true); // 아이템 확장처리만 원함.
11. }
```

static **AXTree.collapseAll**()


 [AXTree.js, line 5227](#)

트리의 모든 아이템을 축소상태로 변경합니다.

Example

```
1. myTree.collapseAll();
```

static **AXTree.expandAll**([depth="all"])

 [AXTree.js, line 5198](#)

트리의 노드를 확장시켜 줍니다.

| Name | Type | Description |
|----------------|------------------------|---|
| [depth="all"]} | String Null Number | 확장할 데프, 값을 주지 않거나 "all" 을 주면 전체 확장이됩니다. |

Example

```
1. myTree.expandAll(); //모두확장
2. myTree.expandAll(1); //1 데프까지만 확장
```

static

[AXTree.js, line 2694](#)

AXTree.expandToggleList(itemIndex, item, expandStat)

아이템의 확장/축소 상태를 토글처리 합니다.

| Name | Type | Description |
|------------|----------|--------------|
| itemIndex | Number | 아이템 인덱스 |
| item | JSObject | 아이템 json |
| expandStat | Boolean | 트리 아이템 오픈 여부 |

Example

```
1. var iwantItemIndex = 10;
2. var myitem = myTree.list[iwantItemIndex];
3. myTree.expandToggleList(iwantItemIndex, myitem);
4. myTree.expandToggleList(iwantItemIndex, myitem, true); // 노드를 열린 상태로 바꾸어 줍니다.
```

static

[AXTree.js, line 1139](#)

AXTree.getCheckedList(colSeq) {Array}

colGroup의 배열순번으로 해당 col의 checked 된 아이템을 반환하여 줍니다.

| Name | Type | Description |
|--------|--------|-----------------------------|
| colSeq | number | checkbox가 있는 colGroup index |


Returns:

된 아이템의 배열

Example

```
1. var myArray = myTree.getCheckedList(0);
```

static AXTree.getSelectedList()
{JSObject}

 [AXTree.js, line 5112](#)

현재 선택된 아이템을 반환합니다. (Number) index of Array 선택한 아이템들의 첫번째

Returns:

item: {} }

Example

```
1. var SL = AXTree.getSelectedList();
2. trace(SL);
```

static AXTree.getSelectedList()
{JSObject}

 [AXTree.js, line 5132](#)

현재 선택된 아이템의 부모 아이템을 반환합니다.

Returns:

item: {} }

Example

```
1. var SL = AXTree.getSelectedListParent();
2. trace(SL);
```

static AXTree.moveTree(Option)

 [AXTree.js, line 4655](#)

원하는 아이템의 위치를 수정합니다.

| Name | Type | Description |
|--------|--------------------------|--|
| Option | JSObject | startMove, validate, endMove, Option에 3가지 함수를 정의합니다. example code 참고 |

Example

```
1. myTree.moveTree({
2.     startMove: function(){           //moveTree가 발동
                                       되었을 때 발생하는 콜백함수
3.         myTree.addClassItem({
4.             className:"disable",
5.             addClass: function(){
6.                 return (this.nodeID == "N");
7.             }
8.         });
9.     },
10.    validate: function(){           //moveTree가 활성화
```

원 상태에서 사용자의 선택을 검증하는 콜백함수

```
11.         //this.moveObj
12.         //this.targetObj
13.         if(this.targetObj.nodeID == "N"){
14.             alert("이동할 수 없는 대상을 선택하셨습니다.");
15.             return false;
16.         }else{
17.             return true;
18.         }
19.     },
20.     endMove: function(){ //moveTree가 완료 되었을때 발생하는 콜백함수
21.         myTree.removeClassItem({
22.             className:"disable",
23.             removeClass:function(){
24.                 return (this.nodeID == "N");
25.             }
26.         });
27.     }
28. });
```

static

 [AXTree.js, line 5173](#)

AXTree.relationFixedSync(options)

{Array}

자식 항목에 체크된 경우 부모 값을 체크된 상태로 변경 해주는 메서드 입니다.

| Name | Type | Description |
|---------|----------|-------------|
| options | JSObject | 설명 |

Returns:

item of list

Example

```
1. myTree.relationFixedSync();
2. myTree.relationFixedSync({expandItem:true}); //
   체크된 아이টে을 확장상태로 변경합니다.
```

static

 [AXTree.js, line 4370](#)

AXTree.removeTree(itemIndex, item)

원하는 아이টে의 데이터를 수정합니다.

| Name | Type | Description |
|-----------|---------------|---|
| itemIndex | Number null | 아이টে index, index는 항목은 null 로 정의해도 처리가 가능합니다. |

| Name | Type | Description |
|------|----------|-------------|
| item | JSObject | 아이템 |

Example

```

1. var obj = myTree.getSelectedList();
2. if(obj.error){
3.     alert("개체를 선택해 주세요");
4.     return;
5. }
6. myTree.removeTree(obj.index, obj.item);

```

static AXTree.setConig(config)

 [AXTree.js, line 948](#)

선언된 클래스를 사용하기 위해 속성을 정의합니다.

| Name | Type | Description |
|--------|--------|-------------|
| config | Object | gridConfig |

Example

```

1. var myTree = new AXTree();
2. myTree.setConfig({
3.     targetID : "AXTreeTarget",  //{String} -
HTML 엘리먼트 타겟아이디
4.     theme : "AXTree_none",  //[String] -
("AXTree", "AXTree_none") CSS Class 이름
5.     relation:{  //부모자식 키 정의
6.         parentKey:"pno",  //부모아이디 키
7.         childKey:"no"  //자식아이디 키
8.     },
9.     colGroup: {  //트리 헤드정의
10.         {
11.             key:"nodeName",  //{String} - 컬럼에 매
치될 item 의 키
12.             label:"제목",  //{String} - 컬럼에 표시할
라벨
13.             width:"100%",  //[Number["px", "%"]
= "auto"] - "100%", "500px", "auto"지정하면 트리의 너
비만큼 단일 컬럼의 너비가 자동 맞춤 처리됩니다.
14.             align:"left",  //[String = "left"
[left, center, right]] - 정렬방식 지정
15.             indent:true,  //[Boolean = true]
16.             getIconClass: function(){  //
[Function] - indent 속성 정의된 대상에 아이콘을 지정할 수
있습니다.
17.                 var iconNames = "folder,
AXfolder, movie, img, zip, file, fileTxt,
fileTag".split(/, /g);
18.                 var iconName = "";
19.                 if(this.item.type) iconName =
iconNames[this.item.type];


```

```

20.         return iconName;
21.     },
22.     formatter: function(){ // [Function]
- 컬럼값의 표현형식 각각 화폐표현식, urlDecode,
input.Checkbox, input.radioButton, 사용자 정의 함수
23.         return "
<b>" + this.item.no.setDigit(2) + "</b> : " +
this.item.nodeName + " (" + this.item.writer +
")";
24.     }
25. }
26. },
27. body: {
28.     onclick: function(idx, item){
//[Function] 바디 클릭 이벤트 콜백함수
29.         toast.push(Object.toJSON(item));
30.     },
31.     ondblclick: function(idx, item){
//[Function] 바디 더블클릭 이벤트 콜백함수
32.         toast.push(Object.toJSON(item));
33.     },
34.     oncheck: function(idx, item){
//[Function] 트리 체크박스클릭시 함수연결
35.         toast.push(Object.toJSON(item));
36.     },
37.     onexpand: function(idx, item){
//[Function] 트리 아이템 확장 이벤트 콜백함수
38.         toast.push(Object.toJSON(item));
39.     },
40.     oncontract: function(idx, item){
//[Function] 트리 아이템 축소 이벤트 콜백함수
41.         toast.push(Object.toJSON(item));
42.     },
43.     addClass: function(idx, item){
//[Function] 트리 아이템에 사용자 CSS 클래스를 추가할 수
있는 사용자 함수 추가하려는 클래스명을 return 으로 반환하십시오
44.         toast.push(Object.toJSON(item));
45.     }
46. }
47. });

```

static **AXTree.setFocus(itemIndex)**

 [AXTree.js, line 3721](#)

index 위치로 트리바디의 포커스를 이동하고 선택된 상태로 변경합니다.

| Name | Type | Description |
|-----------|--------|-------------|
| itemIndex | Number | |

Example

```
1. myTree.setFocus(3);
```

트리에 데이터를 전달합니다. 비동기 방식의 경우 직접데이터를 전달하지 않고 데이터의 전달자 정보를 정의하여 처리합니다.

| Name | Type | Description |
|-------------|--------------------------------|---|
| obj | Array Object | example code 참고 |
| positioning | String | <div>optional</div> 특정 자식개체를 지정해서 하위의 자식노드를 업데이트 합니다. |

Example

```
//Array - list Array setConfig 에서 정의한 relation 의 부모, 자식키 값을 이용하여 list형 데이터를 tree형 데이터로 변환하여 트리를 구성합니다.
var List = [
    {no:1, nodeName:"LEVEL 1-1", writer:"tom", type:"0", pno:0},
    {no:11, nodeName:"LEVEL 1-1-1", writer:"tom", type:"0", pno:1},
    {no:2, nodeName:"LEVEL 2-1", writer:"tom", type:"0", pno:0},
    {no:21, nodeName:"LEVEL 2-1-1", writer:"tom", type:"0", pno:2},
    {no:24, nodeName:"LEVEL 2-1-4", writer:"tom", type:"0", pno:2},
    {no:241, nodeName:"LEVEL 2-1-4-1", writer:"tom", type:"0", pno:24},
    {no:2411, nodeName:"LEVEL 2-1-4-1-1", writer:"tom", type:"0", pno:241},
    {no:2412, nodeName:"LEVEL 2-1-4-1-1", writer:"tom", type:"0", pno:241},
    {no:25, nodeName:"LEVEL 2-1-2", writer:"tom", type:"0", pno:2},
    {no:26, nodeName:"LEVEL 2-1-3", writer:"tom", type:"0", pno:2},
    {no:3, nodeName:"LEVEL 3-1", writer:"tom", type:"0", pno:0},
    {no:11, nodeName:"LEVEL 3-1", writer:"tom", type:"0", pno:0}
];
myTree.setList(List);

var AJAXconfigs = {
    ajaxUrl:"loadTree.php", //{String} - AJAX 호출 URL
    ajaxPars:"param1=1&param2=2", //{String} - AJAX 호출 URL 파라미터 (전송은 post 방식으로 이루어집니다.)
    onLoad: function(){ //[Function] - AJAX 호출완료 이벤트 콜백함수
        ...
    }
};
```



```
myTree.setList(AJAXconfigs);
```

static AXTree.setTree(obj)

 [AXTree.js, line 4068](#)

트리에 데이터를 전달합니다. 비동기 방식의 경우 직접데이터를 전달하지 않고 데이터의 전달자 정보를 정의하여 처리합니다.


| Name | Type | Description |
|------|--|---------------------------------|
| obj | Array Object | example code 참고 |

Example

```
//Array - JSObject(tree형)
var Tree = [
    {no:"1", type:"WBS", activity:"WBS 이름", desc:"",
    charger:"", admin:"", docs:"", open:true, subTree:[
        {no:"1.1", type:"phase", activity:"기획 및 설계",
        desc:"M", charger:"최인석", admin:"", docs:"",
        open:true, subTree:[
            {no:"1.1.1", type:"process", activity:"기획
            단계", desc:"M", charger:"최인석", admin:"", docs:"",
            open:true, subTree:[
                {no:"1.1.1.1", type:"activity",
                activity:"요구사항정의", desc:"M", charger:"최인석/PM",
                admin:"홍길동", docs:"[필수]요구사항정의서", open:false,
                subTree:[ ]},
                {no:"1.1.1.2", type:"activity",
                activity:"업무분할", desc:"M", charger:"한승욱/기획",
                admin:"", docs:"[권고]요구사항정의서", open:false, subTree:
                [ ]}
            ]}
        ]}
    ]},
    {no:"9", type:"WBS", activity:"WBS 이름", desc:"",
    charger:"", admin:"", docs:"", open:true, subTree:[ ]}
];
myTree.setTree(Tree);

var AJAXconfigs = {
    ajaxUrl:"loadTree.php", //{String} - AJAX 호출 URL
    ajaxPars:"param1=1&param2=2", //{String} - AJAX
    호출 URL 파라미터 (전송은 post 방식으로 이루어집니다.)
    onLoad: function(){ //[Function] - AJAX 호출완료 이벤
    트 콜백함수
        ...
    }
};
myTree.setTree(AJAXconfigs);
```

static

 [AXTree.js, line 4336](#)

AXTree.updateTree(itemIndex,

item, obj)

원하는 아이템의 데이터를 수정합니다.

| Name | Type | Description |
|-----------|----------|--------------|
| itemIndex | Number | 아이템 인덱스 |
| item | JSObject | 아이템 |
| obj | JSObject | 수정하려는 아이템 내용 |

Example

```
var obj = myTree.getSelectedList();
myTree.updateTree(obj.index, obj.item,
{nodenm:frm.nodeName.value});
// 수정하려는 아이템의 일부 키만 전달 해도 수정이 가능합니다.
```