빌드 방법 && 적용방법

유닉스 기반 OS 에서 git 을 설치합니다.

```
ex) Ubuntu

$ sudo apt-get update
$ sudo apt-get install git
```

소스코드를 내려받습니다.

```
$ git clone https://github.com/SeungpilPark/OG-Tree
```

소스코드를 내려받으면 OG-Tree 폴더가 생성됩니다.

OG-Tree 폴더 안의 src/main/webapp 폴더 안의 모든 내용을 Aras 시스템의 uengine 폴더로 모두 복사하도록 합니다.

```
$ cp -R OG-Tree/src/main/webapp/*  <Path of uengine in Aras system>/
```

소스 수정 및 Configuration

다음 설명할 파일들의 위치는 OG-Tree/src/main/webapp 하위의 위치를 기준으로 설명합니다.

또한 Aras 시스템에서의 uengine 폴더의 위치이기도 합니다.

콘텍스트 메뉴 아이콘 변경

소스코드의 common/icons/ 하위의 아이콘들을 변경하도록 합니다.

directory	file name	extension	desc (파일 설명)
common/icons/	create-ed.svg	svg	콘텍스트 메뉴 ed 생성 아이콘
	create-folder.svg	svg	콘텍스트 메뉴 폴더 생성 아이콘
	delete-item.svg	svg	콘텍스트 메뉴 아이템 삭제 아이콘
	pick-ed.svg	svg	콘텍스트 메뉴 pick ed 아이콘

액티비티, 폴더, ED 및 기타 도형 이미지

소스코드의 common/shape/ 하위의 다음 파일들을 변경하도록 합니다.

directory	file name	extension	desc (파일 설명)
common/shape/	activity.js	js	액티비티 클래스 정의 파일
	ed.js	js	ED 클래스 정의 파일
	folder.js	js	폴더 클래스 정의 파일
	Expander.js	js	폴더 여닫음 클래스 정의 파일
	Lock.svg	svg	Lock 표현 이미지
	secret.svg	svg	Secret 표현 이미지
	selected.svg	svg	Selected (s라벨) 이미지

	check-image.png	png	체크박스 체크시 표현되는 이미지

HTML 수정

소스코드의 / 하위의 다음 파일들을 변경하도록 합니다.

directory	file name	extension	desc (파일 설명)
/	doosanEditor.html	html	에디터 html
	doosanMonitor.html	html	모니터 html
	doosanWorkflow.html	html	워크플로우차트 html

비지니스 로직 스크립트 수정

비지니스 로직 스크립트는 실제적인 개발 내용이 담긴 스크립트 입니다.

directory	file name	extension	desc (파일 설명)
common/	dataController.js	js	아라스 데이터 통신 담당 클래스
	chartRenderer.js	js	워크플로우 차트 HTML 컨트롤 담당 클래스
	chartViewController.js	js	워크플로우 차트 캔버스 렌더링 클래스
	chartState.json	json	워크플로우 차트 스테이터스 정의 파일
	editorRenderer.js	js	에디터/모니터 캔버스 렌더링 클래스
	editorViewController.js	js	에디터/모니터 HTML 컨트롤 담당 클래스
	state.json	json	에디터/모니터 스테이터스 정의 파일

dataController.js

dataController.js 클래스는 Aras 시스템 데이터 통신 담당 클래스입니다.

이 모듈은 스탠드어론 스크립트 모듈이기 때문에, 이 모듈(html) 을 호출한 부모 페이지로부터 aras 객체를 상속받아 데이터 통신에 사용하게 됩니다.

dataController.js 클래스에 대한 별로듸 Configuration 은 없습니다.

다음은 dataController.js 클래스의 정의 및 프로퍼티 목록입니다.

```
/**
 * Aras data Handler
 *
 * @class
 *
 * @param tree 오픈그래프 트리 라이브러리
 * @author <a href="mailto:sppark@uengine.org">Seungpil Park</a>
 */
var DataController = function (tree, chartRenderer, viewController) {

    /**
     * Doosan view controller
     * @type {Doosan} Doosan
     */
    this.viewController = viewController;
    /**
```

```
 * Renderer 객체
 * @type {Tree} OG-Tree
 */
this.tree = tree;

/**
 * 차트 렌더러
 */
this.chartRenderer = chartRenderer;

/**
 * 부모페이지의 aras 객체
 * @type {aras}
 */
this.aras = null;

/**
 * 현재의 워크플로우 아이템
 * @type {Object}
 */
this.thisItem = null;

/**
 * 현재의 워크플로우 아이디
 * @type {String}
 */
this.wfId = null;

/**
 * 현재 워크플로우의 스탠다드 모드 여부
 * @type {String} Y/N
 */
this.stdYN = null;

/**
 * 현재 워크플로우의 프로젝트 아이디
 * @type {String}
 */
this.projectId = null;

/**
 * newIOMInnovator 리턴 객체
 * @type {null}
 */
this.inn = null;

/**
 * 프로젝트, 폴더 5레벨 까지 enable, ed 는 6레벨까지 가능
 * @type {number}
 */
this.prjMaxDepth = 5;
```

```
    /**
     * 스탠다드, 폴더 3레벨 까지 enable, 3레벨 일 경우는 ed 생성 불가
     * @type {number}
     */
    this.stdMaxDepth = 3;

    /**
     * 아이템 타입 Contants
     * @type {{WORKFLOW: string, ACTIVITY: string, FOLDER: string, ED: string,
MAPPING: string}}
     */
    this.TYPE = {
        WORKFLOW: "workflow",
        ACTIVITY: "activity",
        FOLDER: "folder",
        ED: "ed",
        MAPPING: "mapping"
    };

    /**
     * 스테이트 정의가 저장되어 있는 파일을 불러온다.
     */
    var stateJson;
    $.ajax({
        type: 'GET',
        url: 'common/state.json',
        dataType: 'json',
        async: false,
        success: function (data) {
            stateJson = data;
        }
    });
    this.stateJson = stateJson;

    /**
     * 챠트 스테이트 정의가 저장되어 있는 파일을 불러온다.
     */
    var chartStateJson;
    $.ajax({
        type: 'GET',
        url: 'common/chartState.json',
        dataType: 'json',
        async: false,
        success: function (data) {
            chartStateJson = data;
        }
    });
    this.chartStateJson = chartStateJson;

};
```

editorViewController.js

editorViewController.js 클래스는 doosanEditor.html 과 doosanMonitor.html 페이지 로딩시 호출되며, html 구성 및 이벤트처리,
그리고 OG-Tree 와 aras 데이터간의 연계처리를 수행합니다.

별도의 Configuration 은 없습니다.

다음은 editorViewController.js 클래스의 정의 및 프로퍼티 목록입니다.

```
/**
 * EditorViewController html view Handler
 *
 * @class
 *
 * @author <a href="mailto:sppark@uengine.org">Seungpil Park</a>
 */
var EditorViewController = function () {
    /**
     * OG-Tree 클래스
     * @type {Tree}
     */
    this.tree = null;

    /**
     * Aras 클래스
     * @type {Aras}
     */
    this.aras = null;

    /**
     * Dev 모드
     * @type {string}
     */
    this.mode = 'sample'; //random,sample

    this.edbTypes = [
        {
            label: '2D & 3D Drawing',
            name: 'CAD'
        },
        {
            label: 'Document',
            name: 'Document'
        },
        {
            label: 'Data List',
            name: 'DHI_IntelliSheet'
        }
    ];
};
```

editorRenderer.js

editorRenderer.js(OG-Tree) 클래스는 아라스의 데이터를 화면에 트리 모형의 구조로 렌더링하는 클래스입니다.

아라스 데이터를 렌더링 할 수 있는 데이터 구조로 변환하며, 유엔진의 SVG 렌더링 오픈소스인 오픈그래프의 API 를 이용해 화면에 드로잉합니다.

또한, 드로잉 된 요소에 Dom Event 를 등록하며, 이벤트 발생 시 editorViewController.js 클래스로 전달하는 역할도 수행합니다.

아래와 같은 렌더링 관련 Configuration 요소들이 있습니다.

```
/**
 * Open graph Tree Library (OG-Tree)
 *
 * @class
 * @requires OG.*
 *
 * @param {String} container Dom Element Id
 * @author <a href="mailto:sppark@uengine.org">Seungpil Park</a>
 */
var Tree = function (container) {
    .
    .
    .
    this._CONFIG = {
        /**
         * 액티비티를 이동하여 소트시킬 수 있는 여부
         */
        MOVE_SORTABLE: false,
        /**
         * 아이템을 드래그하여 매핑시킬 수 있는 여부
         */
        MAPPING_ENABLE: false,
        /**
         * 폴더 생성 가능 여부
         */
        CREATE_FOLDER: false,
        /**
         * ED 생성 가능 여부
         */
        CREATE_ED: false,
        /**
         * PICK ED 가능 여부
         */
        PICK_ED: false,
        /**
         * 아이템 삭제 가능 여부
         */
        DELETABLE: false,
        /**
         * 라벨 표기 가능 여부
```

```
 */
SHOW_LABEL: true,
/**
 * 캔버스의 마진
 */
DISPLAY_MARGIN: 50,
/**
 * 캔버스 높이
 */
CONTAINER_HEIGHT: 600,
/**
 * 라벨 최소 크기(IE)
 */
LABEL_MIN_SIZE: 300,

/**
 * 라벨 최대 크기(IE)
 */
LABEL_MAX_SIZE: 400,
/**
 * 라벨 최대 글자 크기
 */
LABEL_MAX_LENGTH: 20,

/**
 * Lane Area 관련 Configration
 */
AREA: {
    /**
     * 전체 화면 중 아더 워크플로우의 가로 비율
     */
    LEFT_SIZE_RATE: (5 / 12) - 0.002,
    /**
     * 전체 화면 중 마이 워크플로우 가로 비율
     */
    RIGHT_SIZE_RATE: (7 / 12) + 0.002,
    /**
     * 액티비티 Area 사이즈
     */
    ACTIVITY_SIZE: 120,
    /**
     * Area 의 Bottom Margin
     */
    BOTTOM_MARGIN: 50,
    /**
     * Area 의 Top Margin
     */
    TOP_MARGIN: 30,
    /**
     * 아더 - 액티비티 Area 라벨 및 디스플레이 여부
     */
    lAc: {
```

```
            label: 'Other Activity',
            display: false
        },
        /**
         * 아더 - 아웃 Area 라벨 및 디스플레이 여부
         */
        lOut: {
            label: 'Other Output',
            display: false
        },
        /**
         * 마이 - 인 Area 라벨 및 디스플레이 여부
         */
        rIn: {
            label: 'My Input',
            display: true
        },
        /**
         * 마이 - 액티비티 Area 라벨 및 디스플레이 여부
         */
        rAc: {
            label: 'My Activity',
            center: true,
            display: true
        },
        /**
         * 마이 - 아웃 Area 라벨 및 디스플레이 여부
         */
        rOut: {
            label: 'My Output',
            display: true
        }
    },
    /**
     * Area 의 스타일 관련 Configuration
     */
    AREA_STYLE: {
        /**
         * 아더 - 액티비티 Area Style
         */
        lAc: {
            'fill': 'RGB(246,246,246)',
            'fill-opacity': '1'
        },
        /**
         * 아더 - 아웃 Area Style
         */
        lOut: {
            'fill': 'RGB(246,246,246)',
            'fill-opacity': '1'
        },
        /**
```

```
         * 마이 - 인 Area Style
         */
        rIn: {
            'fill': 'RGB(255,255,255)',
            'fill-opacity': '1'
        },
        /**
         * 마이 - 액티비티 Area Style
         */
        rAc: {
            'fill': 'RGB(255,255,255)',
            'fill-opacity': '1'
        },
        /**
         * 마이 - 아웃 Area Style
         */
        rOut: {
            'fill': 'RGB(246,246,246)',
            'fill-opacity': '1'
        }
    },
    /**
     * 아이템의 사이즈 관련 Configuration
     */
    SHAPE_SIZE: {
        /**
         * 아이템의 너비 (공통)
         */
        COL_SIZE: 50,
        /**
         * 액티비티 가로
         */
        ACTIVITY_WIDTH: 50,
        /**
         * 액티비티 세로
         */
        ACTIVITY_HEIGHT: 50,
        /**
         * 액티비티 라벨 마진
         */
        ACTIVITY_LABEL_MARGIN: 20,
        /**
         * 액티비티 마진
         */
        ACTIVITY_MARGIN: 50,
        /**
         * 폴더 가로
         */
        FOLDER_WIDTH: 40,
        /**
         * 폴더 세로
         */
```

```
        FOLDER_HEIGHT: 40,
        /**
         * 폴더 마진
         */
        FOLDER_MARGIN: 20,
        /**
         * ED 가로
         */
        ED_WIDTH: 30,
        /**
         * ED 세로
         */
        ED_HEIGHT: 30,
        /**
         * ED 마진
         */
        ED_MARGIN: 24,
        /**
         * Expand 버튼과 부모사이의 간격
         */
        EXPANDER_FROM_MARGIN: 10,
        /**
         * Expand 버튼과 자식 사이의 꺾음 부위 간격
         */
        EXPANDER_TO_VERTICE_MARGIN: 10,
        /**
         * Expand 버튼과 자식 사이의 간격
         */
        EXPANDER_TO_MARGIN: 40,
        /**
         * Expand 버튼 가로
         */
        EXPANDER_WIDTH: 14,
        /**
         * Expand 버튼 세로
         */
        EXPANDER_HEIGHT: 14
    },
    /**
     * 디폴트 스타일 Configuration
     */
    DEFAULT_STYLE: {
        /**
         * Blur 처리하는 아이템의 opacity
         */
        BLUR: "0.3",
        /**
         * 연결선 타입
         */
        EDGE: "plain", //bezier || plain,
        /**
         * 매핑 연결선 타입
```

```
             */
            MAPPING_EDGE: "bezier", //bezier || plain,
            /**
             * 아이템 라벨 폰트 사이즈
             */
            FONT_SIZE: 9
        }
    };
    .
    .
    .
};
```

editorRenderer.js 의 _CONFIG 이하의 프로퍼티 들은 렌더링 요소에 관련된 설정 값들입니다.

이 설정값들을 수정하기 위해서는 직접 소스코드를 수정하는 방법이 있고, 다른 방법으로는 editorViewController.js 클래스의 init 메소드에서 다음과 같이 호출하는 방법이 있습니다.

```
ex)

/**
 * Html 페이지가 처음 로딩되었을 때 오픈그래프 트리를 활성화하고, 필요한 데이터를 인티그레이션 한다.
 */
init: function () {
    .
    .
    .
    me.tree.init();

    /**
     * Blur 처리하는 아이템의 opacity 를 변경한다.
     */
    me.tree._CONFIG.DEFAULT_STYLE.BLUR = "0.5"
```

chartViewController.js

chartViewController.js 클래스는 doosanWorkflow.html 페이지 로딩시 호출되며, html 구성 및 이벤트처리, 그리고 OG-Datatable 과 aras 데이터간의 연계처리를 수행합니다.

별도의 Configuration 은 없습니다.

다음은 chartViewController.js 클래스의 정의 및 프로퍼티 목록입니다.

```
/**
 * ChartViewController html view Handler
 *
 * @class
 *
 * @author <a href="mailto:sppark@uengine.org">Seungpil Park</a>
 */
var ChartViewController = function () {
    /**
     * ChartRenderer 클래스
     * @type {Tree}
     */
    this.renderer = null;

    /**
     * Aras 클래스
     * @type {Aras}
     */
    this.aras = null;

    /**
     * Edit 모드
     * @type {string}
     */
    this.editMode = true;


    this.chartStateJson = null;
};
```

chartRenderer.js

chartRenderer.js(OG-DataTable) 클래스는 아라스의 데이터를 화면에 트리 모형의 구조로 렌더링하는 클래스입니다.

아라스 데이터를 렌더링 할 수 있는 데이터 구조로 변환하며, 유엔진의 SVG 렌더링 오픈소스인 오픈그래프의 API 를 이용해 화면에 드로잉합니다.

또한, 드로잉 된 요소에 Dom Event 를 등록하며, 이벤트 발생 시 chartViewController.js 클래스로 전달하는 역할도 수행합니다.

아래와 같은 렌더링 관련 Configuration 요소들이 있습니다.

```
/**
 * Open graph Chart Library (OG-Tree)
 *
 * @class
 * @requires OG.*
 *
 * @param {String} container Dom Element Id
 * @param {ChartViewController} viewController
 * @author <a href="mailto:sppark@uengine.org">Seungpil Park</a>
 */
```

```
var ChartRenderer = function (container, viewController, editMode) {
    this.editMode = editMode;
    this.viewController = viewController;
    this._CONFIG = {
        /**
         * 캔버스 높이
         */
        CONTAINER_HEIGHT: 800,
        CONTAINER_MIN_HEIGHT: 800,
        CONTAINER_MAX_HEIGHT: 1800,

        ACTIVITY_WIDTH: 80,
        ACTIVITY_HEIGHT: 38,
        ARRANGEMENT: 'horizontal',
        ARRANGEMENT_MARGIN: 24,


        /**
         * 커스텀 컬럼 식별자
         */
        CUSTOM_COL_PREFIX: 'customColPrefix',

        /**
         * 라벨 최대 글자 크기
         */
        LABEL_MAX_LENGTH: 25
    };

    this._CONTAINER = $('#' + container);
    this._CONTAINER.css({
        width: '100%',
        height: this._CONFIG.CONTAINER_HEIGHT + 'px',
        border: '1px solid #555'
    });
    // Canvas
    this.canvas = new OG.Canvas(container, [this._CONTAINER.width(),
this._CONFIG.CONTAINER_HEIGHT], 'white');

    if (this.editMode) {
        this.canvas.initConfig({
            selectable: true,
            dragSelectable: true,
            movable: true,
            resizable: true,
            connectable: true,
            selfConnectable: true,
            connectCloneable: true,
            connectRequired: true,
            labelEditable: true,
            groupDropable: true,
            collapsible: true,
            enableHotKey: false,
```

```
            enableContextMenu: true,
            useSlider: false,
            stickGuide: true,
            checkBridgeEdge: true,
            autoHistory: false
        });
    } else {
        this.canvas.initConfig({
            selectable: true,
            dragSelectable: true,
            movable: false,
            resizable: false,
            connectable: false,
            selfConnectable: false,
            connectCloneable: true,
            connectRequired: true,
            labelEditable: true,
            groupDropable: true,
            collapsible: true,
            enableHotKey: false,
            enableContextMenu: true,
            useSlider: false,
            stickGuide: true,
            checkBridgeEdge: true,
            autoHistory: false
        });
        this.canvas._CONFIG.DELETABLE = false;
    }
    this.canvas._CONFIG.DEFAULT_STYLE.EDGE = {
        stroke: "black",
        fill: "none",
        "fill-opacity": 0,
        "stroke-width": 1,
        "stroke-opacity": 1,
        "edge-type": "plain",
        "arrow-start": "none",
        "arrow-end": "block",
        "stroke-dasharray": "",
        "label-position": "center",
        "stroke-linejoin": "round",
        cursor: "pointer"
    };
    this.canvas._CONFIG.GUIDE_CONTROL_LINE_NUM = 1;
    this.canvas._CONFIG.DRAG_PAGE_MOVABLE = true;
    this.canvas._CONFIG.FOCUS_CANVAS_ONSELECT = false;
    this.canvas._CONFIG.SPOT_ON_SELECT = true;
    this.canvas._CONFIG.STICK_GUIDE = false;
    this.canvas._CONFIG.AUTOMATIC_GUIDANCE = false;

    this._RENDERER = this.canvas._RENDERER;
    this._HANDLER = this.canvas._HANDLER;
```

```
    this.existJson = null;
    this.loadElements = null;
    this.existActivitySize = {};


    /**
     * 최종 데이터 테이블
     * @type {null}
     */
    this.dataTable = null;
    /**
     * 최종 로우데이터
     * @type {Array}
     */
    this.finalRowData = [];
    /**
     * 재구성될 커넥션 리스트
     * @type {Array}
     */
    this.connections = [];
    /**
     * 재구성 된 커넥션 edgeId 리스트
     * @type {Array}
     */
    this.existConnections = [];
    /**
     * 뷰 모드일시 그려진 셀 리스트
     * rowIndex + '-' + column 으로 표현한다.
     * @type {Array}
     */
    this.existCells = [];

};
```

디버깅 툴 및 사용법

OG-Tree 모듈은 html 과 js 파일로만 이루어져 있습니다.

따라서 디버깅은 브라우저의 개발자 도구(F12) 를 통해 수행할 수 있습니다.

구성 가이드

전체 모듈의 구성 요소에 대한 표입니다.

분류	directory	file name	extension	desc (파일 설명)
콘텍스트 메뉴 아이콘	common/icons/	create-ed.svg	svg	콘텍스트 메뉴 ed 생성 아이콘
		create-folder.svg	svg	콘텍스트 메뉴 폴더 생성 아이콘
		delete-item.svg	svg	콘텍스트 메뉴 아이템 삭제 아이콘

		pick-ed.svg	svg	콘텍스트 메뉴 pick ed 아이콘
개발용 샘플 데이터	common/sample/	myData.json	json	개발용 샘플 – 마이 워크플로우 데이터
		otherData.json	json	개발용 샘플 – 아더 워크플로우 데이터
		chartData.json	json	개발용 샘플 – 워크플로우 차트 데이터
		chartMap.json	json	개발용 샘플 – 워크플로우 차트 맵 데이터
도형 클래스 및 이미지	common/shape/	Activity.js	js	액티비티 – 오픈그래프 클래스 파일
		Area.js	js	Area – 오픈그래프 클래스 파일
		Expander.js	js	폴더 열림/닫힘 버튼 – 오픈그래프 클래스 파일
		Ed.js	js	ED – 오픈그래프 클래스 파일
		Folder.js	js	폴더 – 오픈그래프 클래스 파일
		SLabel.js	js	셀렉티트 표기 – 오픈그래프 클래스 파일
		selected.svg	svg	셀렉티트 표기 – svg 이미지 파일
		Lock.svg	svg	Lock 표현 이미지
		secret.svg	svg	Secret 표현 이미지
		check-image.png	png	체크박스 체크시 표현되는 이미지
라이브러리	lib/	jquery-1.11.1	directory	Jquery 라이브러리
		jquery-contextMenu	directory	Jquery 콘텍스트 메뉴 라이브러리
		jquery-ui-1.11.0.custom	directory	Jquery UI 라이브러리
		jqueryXML2JSON	directory	Jquery xml to json 변환 라이브러리
		opengraph	directory	오픈그래프 라이브러리
부트스트랩 resources	resources/	*	directory	부트스트랩 디렉토리
HTML	/	doosanEditor.html	html	에디터 html
		doosanMonitor.html	html	모니터 html
		doosanWorkflow.html	html	워크플로우 차트 html
비지니스 로직 스크립트	common/	dataController.js	js	아라스 데이터 통신 담당 클래스
		chartRenderer.js	js	워크플로우 차트 HTML 컨트롤 담당 클래스
		chartViewController.js	js	워크플로우 차트 캔버스 렌더링 클래스
		chartState.json	json	워크플로우 차트 스테이터스 정의 파일
		editorRenderer.js	js	에디터/모니터 캔버스 렌더링 클래스
		editorViewController.js	js	에디터/모니터 HTML 컨트롤 담당 클래스
		state.json	json	에디터/모니터 스테이터스 정의 파

일