//Lake.js部分，主模块部分

var lakeLasttime;

var lakeDeltatime;

//动画效果

var lakelastTime = 0;

var lake\_slide\_views=[];//需要应用动画的全局变量

var lakeslideViewAnim;

var lake\_navs=[];

var lake\_chartPies=[];

var lake\_chartRings=[];

var lake\_chartHistograms=[];

var lake\_chartSins = [];

var lake\_lineAnims=[];

var lake\_colorBgs = [];

var requestAnimationFrame = window.requestAnimationFrame || window.mozRequestAnimationFrame || window.webkitRequestAnimationFrame || window.msRequestAnimationFrame;

var cancelAnimationFrame = window.cancelAnimationFrame || window.mozCancelAnimationFrame;

if ( !requestAnimationFrame || !cancelAnimationFrame ) {

requestAnimationFrame = function( callback, element ) {

var currTime = new Date().getTime();

//为了使setTimteout的尽可能的接近每秒60帧的效果

var timeToCall = Math.max( 0, 16 - ( currTime - lakelastTime ) );

var id = window.setTimeout( callback, timeToCall );

lakelastTime = currTime + timeToCall;

return id;

};

cancelAnimationFrame = function( id ) {

window.clearTimeout( id );

};

}

function lakeanimate(){

var now = Date.now();

lakeDeltatime = now - lakeLasttime;

lakeLasttime = now;

for(var i=0;i<lake\_slide\_views.length;i++){

lake\_slide\_views[i].slide();//滑动视图函数

}

for (var i = 0; i < lake\_navs.length; i++) {

lake\_navs[i].adapt();

}

for (var i = 0; i < lake\_chartPies.length; i++) {

//lake\_chartPies[i].init();

lake\_chartPies[i].draw();

}

for (var i = 0; i < lake\_chartRings.length; i++) {

lake\_chartRings[i].draw();

};

for (var i = 0; i < lake\_lineAnims.length; i++) {

lake\_lineAnims[i].anim();

};

for(var i = 0; i < lake\_chartHistograms.length; i++){

lake\_chartHistograms[i].draw();

};

for (var i = 0; i < lake\_chartSins.length; i++) {

lake\_chartSins[i].draw();

};

lakeslideViewAnim=requestAnimationFrame(lakeanimate);

}

function loadScript(url,callback){

var script = document.createElement("script");

script.type = "text/javascript";

if(script.readyState){//IE

script.onreadystatechange = function(){

if(script.readyState == "loaded" || script.readyState == "complete"){

script.onreadystatechange = null;

callback();

}

}

}else{

script.onload = function(){

callback();

}

}

script.src = url;

console.log("url:"+url);

var scripts = document.getElementsByTagName("head")[0].getElementsByTagName("script");

for (var i = 0; i < scripts.length; i++) {

console.log("script.src:"+scripts[i].src);

if(scripts[i].src == 'http://localhost:3000'+url){

console.log("存在相同sr");

scripts[i].remove();

}

};

document.getElementsByTagName("head")[0].appendChild(script);

}//添加src="url"的<script>标签并load

window.onload=function(){

lakeLasttime = Date.now();

uiResizablePanelLoad();

colorButtonLoad();

slideViewLoad();

navLoad();

chartLoad();

chartSinLoad();

chartHistogramsLoad();

chartRingsLoad();

chartPiesLoad();

chartLinesLoad();

scaleLargeLoad();

sideBarLoad();

lineAnimLoad();

addLoad();

lakeanimate();//load slideView的动画

}

function colorBgLoad(){

var colorBgs = document.getElementsByClassName("lakeColorBg");

if(colorBgs.length!=0){

loadScript("/js/colorBg.js",function(){

lake\_colorBgs = [];

for (var i = 0; i < colorBgs.length; i++) {

lake\_colorBgs[i] = new colorBgObj();

lake\_colorBgs[i].init(colorBgs[i]);

lake\_colorBgs[i].draw();

};

})

}

}

function addLoad(){

var adds = document.getElementsByClassName("lakeAdd");

if(adds.length!=0){

loadScript("/js/add.js",function(){

var lake\_adds = [];

for (var i = 0; i < adds.length; i++) {

lake\_adds[i] = new addObj();

lake\_adds[i].init(adds[i]);

};

});

}

}

function lineAnimLoad(){

var lineAnims = document.getElementsByClassName("lakeLineAnim");

if(lineAnims.length!=0){

loadScript("/js/lineAnim.js",function(){

for (var i = 0; i < lineAnims.length; i++) {

lake\_lineAnims[i] = new lineAnimObj();

lake\_lineAnims[i].init(lineAnims[i]);

// lake\_lineAnims[i].anim();

};

})

}

}

function chartLoad(){

loadScript("/js/charts/chartCoordinate.js",function(){});

loadScript("/js/charts/chartRect.js",function(){});

}

function chartSinLoad(){

var chartSins = document.getElementsByClassName("lakeChartSin");

if (chartSins.length!=0) {

loadScript("/js/charts/chartSin.js",function(){

lake\_chartSins = [];

for (var i = 0; i < chartSins.length; i++) {

lake\_chartSins[i] = new chartSinObj();

lake\_chartSins[i].init(chartSins[i]);

lake\_chartSins[i].draw();

};

});

};

}

function chartHistogramsLoad(){

var chartHistograms = document.getElementsByClassName("lakeChartHistogram");

if(chartHistograms.length!=0){

loadScript("/js/charts/chartHistogram.js",function(){

for (var i = 0; i < chartHistograms.length; i++) {

lake\_chartHistograms[i] = new chartHistogramObj();

lake\_chartHistograms[i].init(chartHistograms[i]);

//lake\_chartHistograms[i].draw();

};

});

}

}

function chartRingsLoad(){

var chartRings = document.getElementsByClassName("lakeChartRing");

if (chartRings.length!=0) {

loadScript("/js/charts/chartRing.js",function(){

loadScript("/js/charts/chartRingPart.js",function(){

lake\_chartRings = [];

for (var i = 0; i < chartRings.length; i++) {

lake\_chartRings[i] = new chartRingObj();

lake\_chartRings[i].init(chartRings[i]);

lake\_chartRings[i].draw();

};

})

})

}//判断环形，导入chartRing.js

}

function chartPiesLoad(){

var chartPies = document.getElementsByClassName("lakeChartPie");

if (chartPies.length!=0) {

loadScript("/js/charts/chartPie.js",function(){

loadScript("/js/charts/chartPiePart.js",function(){

lake\_chartPies = [];

for (var i = 0; i < chartPies.length; i++) {

lake\_chartPies[i] = new chartPieObj();

lake\_chartPies[i].init(chartPies[i]);

lake\_chartPies[i].draw();

};

})

});

}//判断扇形，导入chartPie.js

}

function chartLinesLoad(){

var chartLines = document.getElementsByClassName("lakeChartLine");

if (chartLines.length!=0) {

loadScript("/js/charts/chartLine.js",function(){

var lake\_chartLines = [];

for (var i = 0; i < chartLines.length; i++) {

lake\_chartLines[i] = new chartLineObj();

lake\_chartLines[i].init(chartLines[i]);

};

})

};//判断折线图,导入chartLine.js

}

function sideBarLoad(){

var sideBars = document.getElementsByClassName("lakeSideBar");

if(sideBars.length!=0){

loadScript("/js/sideBar.js",function(){

var lake\_sideBars = [];

for (var i = 0; i < sideBars.length; i++) {

lake\_sideBars[i] = new sideBarObj();

lake\_sideBars[i].init(sideBars[i]);

};

})

}

}

function navLoad(){

var navs = document.getElementsByClassName("lakeNav");

if (navs.length!=0) {

loadScript("/js/nav.js",function(){

for (var i = 0; i < navs.length; i++) {

lake\_navs[i]=new navObj();

lake\_navs[i].init(navs[i]);

};

})

};

}

//判断页面中是否存在id为uiResizablePanel的元素，如果有init

function uiResizablePanelLoad(){

var uiResizablePanel=document.getElementById("uiResizablePanel");

if(uiResizablePanel){

loadScript("/js/resizablePanel.js",function(){

var resizablePanel=new resizablePanelObj();

resizablePanel.init();

});

}

}

//判断页面是否存在className为"colorButton"的元素，如果有,init()，支持多个同时存在

function colorButtonLoad(){

var colorButtons=document.getElementsByClassName("lakeColorButton");

var len=colorButtons.length;

if(len!=0){

loadScript("/js/colorButton.js",function(){

var buttons=[];

for(var i=0;i<len;i++){

buttons[i]=new colorButtonObj();

buttons[i].init(colorButtons[i]);

}

});

}

}

function scaleLargeLoad(){

var scaleLarges = document.getElementsByClassName("lakeScaleLarge");

var len = scaleLarges.length;

if(len!=0){

loadScript("/js/scaleLarge.js",function(){

var lake\_scaleLarges = [];

for (var i = 0; i < scaleLarges.length; i++) {

lake\_scaleLarges[i]=new scaleLargeObj();

lake\_scaleLarges[i].init(scaleLarges[i]);

};

})

}

}

//loadScript("js/slideView.js",function(){});

function slideViewLoad(){

var slideViews=document.getElementsByClassName("lakeSlideView");

var len=slideViews.length;

if (len!=0) {

loadScript("/js/slideView.js",function(){

for (var i=0;i<len;i++) {

lake\_slide\_views[i]=new slideViewObj();

lake\_slide\_views[i].init(slideViews[i]);

lake\_slide\_views[i].canvasPre.addEventListener("click",function(){

for(var i=0;i<lake\_slide\_views.length;i++){

lake\_slide\_views[i].counter = changeView(lake\_slide\_views[i].counter,lake\_slide\_views[i].len);

showView(lake\_slide\_views[i].counter,lake\_slide\_views[i].divs,lake\_slide\_views[i].len);

lake\_slide\_views[i].timer = 0;

}

})

lake\_slide\_views[i].canvasPre.addEventListener("mouseover",function(){

for(var i=0;i<lake\_slide\_views.length;i++){

drawPreHover(lake\_slide\_views[i].canvasPre);

}

})

lake\_slide\_views[i].canvasPre.addEventListener("mouseout",function(){

for(var i=0;i<lake\_slide\_views.length;i++){

drawPre(lake\_slide\_views[i].canvasPre);

}

})

for (var j = 0; j < lake\_slide\_views[i].roundLi.length; j++) {

lake\_slide\_views[i].roundLi[j].addEventListener("mousedown",function(){

for(var i=0;i<lake\_slide\_views.length;i++){

for (var a = 0; a < lake\_slide\_views[i].roundLi.length; a++) {

lake\_slide\_views[i].roundLi[a].onclick = function(){

changeViewByClick(this.id);

}

}

}

})

}

}//for()

});//loadScript

}//if(len!=0)

}//slideViewLoad

//slideView.js 幻灯片控件实现模块

var slideViewObj = function(){

this.counter;

this.timer;

this.intervaller;

this.len;

this.divs=[];

this.canvasPre;

this.roundUl;

this.roundLi=[];

this.update = false;

}

slideViewObj.prototype.init=function(ele){

this.counter = 0;

this.timer = 0;

this.intervaller = 4000;

this.len = 0;

//this.pause=false;

this.divs=ele.getElementsByTagName("span");

this.len=this.divs.length;

var canvasPreGlobalAlpha = 0;

var data\_interval = ele.getAttribute("data-interval");

if(data\_interval){

this.intervaller=data\_interval;

}

var srcs=[];

for (var i = 0; i < this.len; i++) {

srcs[i]=this.divs[i].getAttribute("data-src");

this.divs[i].style.background = "url("+srcs[i]+") no-repeat";

if(i == 0){

this.divs[i].style.opacity = "1";

}else{

this.divs[i].style.opacity = "0";

}

this.divs[i].addEventListener("mouseover",function(){

cancelAnimationFrame(lakeslideViewAnim);//鼠标移入取消/暂停动画切换效果

});

this.divs[i].addEventListener("mouseout",function(){

requestAnimationFrame(lakeanimate);//鼠标移出开始执行计算动画

});

};

if(!this.update){

this.canvasPre = document.createElement("canvas");

this.canvasPre.width = 75;

this.canvasPre.height = 150 ;

//怎样改变元素位置，js? //可以用css解决，下面有css类名

this.canvasPre.className = "preCanvas";

//drawPre(this.canvasPre);

this.roundUl = document.createElement("ul");

this.roundUl.className = "roundList";

for (var i = 0; i < this.len; i++) {

this.roundLi[i]=document.createElement("li");

this.roundLi[i].setAttribute("id","slideViewRoundLi"+i);

this.roundUl.appendChild(this.roundLi[i]);

};

this.roundLi[0].style.transform = "scale(1)";

var canvas = ele.getElementsByTagName('canvas');

var uls = ele.getElementsByTagName('ul');

if (canvas.length >= 1) {

canvas[0].remove();

uls[0].remove();

};

ele.appendChild(this.canvasPre);

ele.appendChild(this.roundUl);

}

//设置next/pre click

}

slideViewObj.prototype.slide=function(){

this.timer = lakeDeltatime + this.timer;

if(this.timer > this.intervaller){

this.counter=changeView(this.counter,this.len);

this.divs = showView(this.counter,this.divs,this.len);

this.timer %= this.intervaller;

}

// console.log(this.counter);

for(var i=0;i<this.len;i++){

if(i == this.counter){

this.roundLi[this.counter].style.transform="scale(1)";

}else{

this.roundLi[i].style.transform="scale(0.6)";

}

}

}

//改变视图，通过counter

function changeView(counter,l){

if(counter == parseInt(l) - 1){

counter = 0;

}else{

counter = parseInt(counter) + 1;

}

return counter;

}

//显示视图,设置DOM

function showView(c,d,l){

for (var i = 0; i < l; i++) {

if(i == c){

d[i].style.opacity = "1";

}else{

d[i].style.opacity = "0";

}

};

return d;

}

function changePause(p){

if(p == false){

p = true;

}else{

p = false;

}

return p;

}

function changeViewByClick(id){

id=id.slice(-1);

for (var i = 0; i < lake\_slide\_views.length; i++) {

lake\_slide\_views[i].counter=id;

showView(lake\_slide\_views[i].counter,lake\_slide\_views[i].divs,lake\_slide\_views[i].len);//ex:i=slideViewRoundLi3

console.log(lake\_slide\_views[i].counter);

lake\_slide\_views[i].timer = 0;

};

}

function drawPre(c){

var canvasPWidth = c.width;

var canvasPHeight = c.height;

var canvasPContext = c.getContext("2d");

canvasPContext.clearRect(0,0,canvasPWidth,canvasPHeight);

canvasPContext.save();

canvasPContext.globalAlpha = 0.6;

canvasPContext.lineWidth = 6;

canvasPContext.strokeStyle = "white";

canvasPContext.shadowBlur = 8;

canvasPContext.shadowColor = "white";

canvasPContext.lineCap = "round";

canvasPContext.lineJoin = "round";

canvasPContext.moveTo(canvasPWidth/4,canvasPHeight/4);

canvasPContext.lineTo(canvasPWidth/4\*3,canvasPHeight/2);

canvasPContext.lineTo(canvasPWidth/4,canvasPHeight/4\*3);

canvasPContext.stroke();

canvasPContext.restore();

}

function drawPreHover(c){

var canvasPWidth = c.width;

var canvasPHeight = c.height;

var canvasPContext = c.getContext("2d");

canvasPContext.clearRect(0,0,canvasPWidth,canvasPHeight);

canvasPContext.save();

canvasPContext.globalAlpha = 1;

canvasPContext.lineWidth = 6;

canvasPContext.strokeStyle = "white";

canvasPContext.shadowBlur = 30;

canvasPContext.shadowColor = "white";

canvasPContext.lineCap = "round";

canvasPContext.lineJoin = "round";

canvasPContext.moveTo(canvasPWidth/4,canvasPHeight/4);

canvasPContext.lineTo(canvasPWidth/4\*3,canvasPHeight/2);

canvasPContext.lineTo(canvasPWidth/4,canvasPHeight/4\*3);

canvasPContext.stroke();

canvasPContext.restore();

}

//scaleLarge.js 放大镜控件实现模块

var scaleLargeObj=function(){

//this.image;

}

scaleLargeObj.prototype.init=function(ele){

var data\_image = ele.getAttribute("data-image");

var canvasScaleLarge = document.createElement("canvas");

var canvasScaleShow = document.createElement("canvas");

canvasScaleLarge.className = "lakeScaleMirror";

canvasScaleLarge.style.display = 'none';

ele.appendChild(canvasScaleShow);

ele.appendChild(canvasScaleLarge);

var canvasScaleShowContext = canvasScaleShow.getContext("2d");

var canvasScaleLargeContext = canvasScaleLarge.getContext("2d");

var image = new Image();

console.log(data\_image);

image.src = data\_image;

image.onload = function(){

var imageWidth = image.width;

var imageHeight = image.height;

var wBH = imageWidth/imageHeight;

var canShowWidth = 400;

var canShowHeight = canShowWidth/wBH;

canvasScaleShow.width = canShowWidth;

canvasScaleShow.height = canShowHeight;

canvasScaleLarge.width = 250;

canvasScaleLarge.height = 250;

var canLargeWidth = canvasScaleLarge.width;

var canLargeHeight = canvasScaleLarge.height;

drawScaleShow(canvasScaleShowContext,image,canShowWidth,canShowHeight);

canvasScaleShow.addEventListener("mousemove",function(e){

this.parentNode.getElementsByClassName('lakeScaleMirror')[0].style.display = 'block';

var mouseX = e.offsetX;

var mouseY = e.offsetY;

//mouseX/canShowWidth = ?/image.width

var originX = (mouseX)/canShowWidth \* image.width;

var originY = (mouseY)/canShowHeight \* image.height;

drawScaleLarge(canvasScaleLargeContext,image,canLargeWidth,canLargeHeight,originX,originY);

})

canvasScaleShow.addEventListener("mouseout",function(){

this.parentNode.getElementsByClassName('lakeScaleMirror')[0].style.display = 'none';

clearScaleLarge(canvasScaleLargeContext,canLargeWidth,canLargeHeight);

})

}

}

function drawScaleShow(ctx,image,cw,ch){

ctx.save();

ctx.drawImage(image,0,0,cw,ch);

ctx.restore();

}

function drawScaleLarge(ctx,image,cw,ch,ox,oy){

ctx.clearRect(0,0,cw,ch);

ctx.save();

ctx.drawImage(image,ox-100,oy-100,200,200,0,0,cw,ch);

ctx.restore();

}

function clearScaleLarge(ctx,cw,ch){

ctx.clearRect(0,0,cw,ch);

}

//resizablePanel.js可变容器控件实现模块

var resizablePanelObj=function(){

}

resizablePanelObj.prototype.init=function(){

resizablePanel("uiResizablePanel");

}

function mouseDownResizablePanel(e,panel,ctrl,type){

var e = e||window.event;

//鼠标在当前元素的位置

m\_start\_x = e.pageX - ctrl.offsetLeft;

m\_start\_y = e.pageY - ctrl.offsetTop;

m\_panel = panel;

m\_ctrl = ctrl;

m\_type = type;

//开始侦听移动

moving = setInterval(mouseMoveResizablePanel,10);

}

document.onmousemove = function(e){

var e=window.event||e;

m\_to\_x = e.pageX;

m\_to\_y = e.pageY;

}

function mouseUpResizablePanel(){

clearInterval(moving);

moving = 0;

var cls = document.getElementsByClassName('ui-Resizable-ctrl');

for(var i = 0;i < cls.length;i++){

cls[i].style.left = "";

cls[i].style.top = "";

}

}

function mouseMoveResizablePanel(){

if(moving){

var to\_x = m\_to\_x - m\_start\_x;

var to\_y = m\_to\_y - m\_start\_y;

var min\_x = m\_panel.offsetLeft;

var min\_y = m\_panel.offsetTop;

to\_x=Math.max(min\_x,to\_x);

to\_y=Math.max(min\_y,to\_y);

switch(m\_type){

case 'r':

m\_ctrl.style.left = to\_x + 'px';

m\_panel.style.width = to\_x +10+ 'px';

break;

case 'b':

//m\_ctrl.style.left = to\_x + 'px';

m\_ctrl.style.top = to\_y + 'px';

m\_panel.style.height = to\_y +10+ 'px';

break;

case 'rb':

m\_ctrl.style.left = to\_x + 'px';

m\_ctrl.style.top = to\_y + 'px';

m\_panel.style.width = to\_x +20+ 'px';

m\_panel.style.height = to\_y +20+ 'px';

break;

}

}

}

function resizablePanel(panel\_id){

var panel=document.getElementById(panel\_id);

var r=document.createElement("div");

var b=document.createElement("div");

var rb=document.createElement("div");

r.class=r.className="ui-Resizable-r ui-Resizable-ctrl";

b.class=b.className="ui-Resizable-b ui-Resizable-ctrl";

rb.class=rb.className="ui-Resizable-rb ui-Resizable-ctrl";

panel.appendChild(r);

panel.appendChild(b);

panel.appendChild(rb);

r.addEventListener('mousedown',function(e){

mouseDownResizablePanel(e,panel,r,'r');

});

b.addEventListener('mousedown',function(e){

mouseDownResizablePanel(e,panel,b,'b');

})

rb.addEventListener('mousedown',function(e){

mouseDownResizablePanel(e,panel,rb,'rb');

})

r.addEventListener('mouseup',function(e){

mouseUpResizablePanel();

});

b.addEventListener('mouseup',function(e){

mouseUpResizablePanel();

});

rb.addEventListener('mouseup',function(e){

mouseUpResizablePanel();

});

}

//nav.js导航条控件实现模块

var navObj=function(){

this.len;

this.nav;

this.nav\_items=[];

this.canvasItems;

this.canvasItemsMask;

}

navObj.prototype.init=function(ele){

var dbWidth=document.body.clientWidth;

this.nav\_items=ele.getElementsByTagName("span");

this.len = this.nav\_items.length;

this.nav = ele;

this.canvasItemsMask = 0;

var canvasItems = document.createElement("canvas");

this.canvasItems=canvasItems;

canvasItems.className = "lakeItems";

canvasItems.width = 50;

canvasItems.height = 50;

canvasItems.style.zIndex = 5;

var canvasWidth = canvasItems.width;

var canvasHeight = canvasItems.height;

//canvasItems.setAttribute("id","canvas\_items");

var canvasItemsContext=canvasItems.getContext("2d");

canvasItemsContext.save();

canvasItemsContext.clearRect(0,0,canvasWidth,canvasHeight);

canvasItemsContext.lineWidth = 4;

canvasItemsContext.strokeStyle = "white";

canvasItemsContext.lineCap = "round";

canvasItemsContext.globalAlpha = 0.8;

canvasItemsContext.moveTo(canvasWidth/4,canvasHeight/4);

canvasItemsContext.lineTo(canvasWidth/4\*0.5\*3,canvasHeight/4);

canvasItemsContext.moveTo(parseInt(canvasWidth/2),canvasHeight/4);

canvasItemsContext.lineTo(canvasWidth/4\*3.5,canvasHeight/4);

canvasItemsContext.moveTo(canvasWidth/4,canvasHeight/4\*2);

canvasItemsContext.lineTo(canvasWidth/4\*0.5\*3,canvasHeight/4\*2);

canvasItemsContext.moveTo(parseInt(canvasWidth/2),canvasHeight/4\*2);

canvasItemsContext.lineTo(canvasWidth/4\*3.5,canvasHeight/4\*2);

canvasItemsContext.moveTo(canvasWidth/4,canvasHeight/4\*3);

canvasItemsContext.lineTo(canvasWidth/4\*0.5\*3,canvasHeight/4\*3);

canvasItemsContext.moveTo(parseInt(canvasWidth/2),canvasHeight/4\*3);

canvasItemsContext.lineTo(canvasWidth/4\*3.5,canvasHeight/4\*3);

canvasItemsContext.stroke();

canvasItemsContext.restore();

document.body.appendChild(canvasItems);

canvasItems.addEventListener("mousedown",function(){

if (lake\_navs[0].canvasItemsMask == 0) {

lake\_navs[0].nav.style.transform = "scale(1)";

for (var i = 0; i < lake\_navs[0].nav\_items.length; i++) {

lake\_navs[0].nav\_items[i].style.padding = "0px";

};

lake\_navs[0].canvasItemsMask = 1;

}else{

lake\_navs[0].nav.style.transform = "scale(0.001)";

lake\_navs[0].canvasItemsMask = 0;

};

})

}

navObj.prototype.adapt=function(){

if (parseInt(document.body.clientWidth) > 600) {

this.canvasItems.style.display = "none";

var dbWidth=document.body.clientWidth;

var nav\_items\_paddingLeft = dbWidth/this.len \* 0.2;

for (var i = 0; i < this.nav\_items.length; i++) {

this.nav\_items[i].style.paddingLeft = nav\_items\_paddingLeft/2+"px";

this.nav\_items[i].style.paddingRight = nav\_items\_paddingLeft/2+"px";

}

lake\_navs[0].nav.style.transform = "scale(1)";

}else{

this.canvasItems.style.display = "block";

this.nav.style.paddingLeft = "0px";

for (var i = 0; i < this.nav\_items.length; i++) {

this.nav\_items[i].style.paddingLeft = 0+"px";

this.nav\_items[i].style.paddingRight = 0+"px";

}

};

}

//colorButton.js按钮控件实现模块

var colorButtonObj=function(){

this.unit=20;

}

colorButtonObj.prototype.init=function(ele){

var data\_color=ele.getAttribute("data-color");

var data\_size=ele.getAttribute("data-size");

var data\_name=ele.getAttribute("data-name");

data\_size = Math.min(10,data\_size);

data\_size = Math.max(1.8,data\_size);

if(data\_size!=null){

var divWidth=parseInt(this.unit\*data\_size\*2);

var divHeight=parseInt(this.unit\*data\_size-divWidth/10);

}else{

var divWidth=parseInt(this.unit\*2\*2);

var divHeight=parseInt(this.unit\*2-divWidth/10);

}

var divFontSize = divHeight / 2;

var divLineHeight = (((divHeight - divFontSize) / 2) / divFontSize)\*3.6;

ele.style.width = divWidth + "px";

ele.style.height = divHeight + "px";

ele.style.lineHeight = divLineHeight.toString();

ele.className = "lakeColorButton colorButtonClass";

ele.style.fontSize = divFontSize + "px";

}

//chartCoordinate.js坐标轴实现模块

//这是一个坐标轴对象

var chartCoordinateObj = function(){

this.canvas;

this.dataX=[];

this.dataY=[];

this.unitPxX;

this.xWidth;

this.yWidth;

this.lineX = [];

this.originX;

this.originY;

}

chartCoordinateObj.prototype.init = function(){

var canvas = this.canvas;

var context = canvas.getContext("2d");

var canWidth = canvas.width;

var canHeight = canvas.height;

this.originX = 20;

this.originY = canHeight - 65;

var xEndX = canWidth - 10;

var yEndY = 40;//this.originY - 150;

this.xWidth = xEndX - this.originX;

this.yWidth = this.originY - yEndY;

yWidth = this.yWidth;

var dataX = this.dataX;

var dataY = this.dataY;

var dataXQuantites = dataX.length;

this.unitPxX = this.xWidth/dataXQuantites;

var unitPxX = this.unitPxX;

var unitPxY = this.yWidth/dataY.length;

this.lineX = [];

var lineY = [];

var textX = [];

for (var i = 0; i < dataXQuantites; i++) {

this.lineX[i] = this.originX + unitPxX\*(i+1);

textX[i] = this.lineX[i] - unitPxX/2 - 8; //here

};

for (var j = 0; j < (yWidth/unitPxY); j++) {

lineY[j] = this.originY - unitPxY\*(j+1);

};

drawCoordinate(context,this.originX,this.originY,xEndX,yEndY,this.lineX,lineY,dataX,dataY,textX,canWidth,canHeight);

}

function drawCoordinate(ctx,ox,oy,xex,yey,lx,ly,dx,dy,tx,cw,ch){

ctx.clearRect(0,0,cw,ch);

ctx.save();

ctx.strokeStyle = "#CFCFCF";

ctx.lineWidth = 1;

ctx.beginPath();

ctx.moveTo(ox,oy);

ctx.lineTo(xex,oy);

ctx.moveTo(ox,oy);

ctx.lineTo(ox,yey);

ctx.stroke();

ctx.save();

ctx.fillStyle = "#8B8386";

ctx.font = "12px Microsoft Yahei";

ctx.fillText("0",ox-20,oy);

ctx.restore();

ctx.save();

ctx.strokeStyle = "#CFCFCF";

ctx.lineWidth = 1;

ctx.fillStyle = "#8B8386";

ctx.font = "12px Microsoft Yahei";

for (var i = 0; i < lx.length; i++) {

ctx.moveTo(lx[i],oy);

ctx.lineTo(lx[i],yey);

ctx.fillText(dx[i],tx[i],oy+20);

};

for (var j = 0; j < ly.length; j++) {

ctx.moveTo(ox,ly[j]);

ctx.lineTo(xex,ly[j]);

};

ctx.stroke();

ctx.restore();

}

//chartLine.js折线图实现模块

var chartLineObj = function(){

this.chartLine;

this.dataX = [];

this.dataY = [];

this.dataXLength = [];

this.dataArcSize;

this.nameX;

this.nameY;

this.titleName;

}

chartLineObj.prototype.init = function(ele){

this.chartLine = ele;

var chartDataX = ele.getAttribute("data-chartDataX");

var chartDataY = ele.getAttribute("data-chartDataY");

this.nameX = ele.getAttribute("data-nameX");

this.nameY = ele.getAttribute("data-nameY");

this.titleName = ele.getAttribute("data-title");

this.dataX = chartDataX.split(",");

this.dataY = chartDataY.split(",");

this.dataArcSize = 3;

//console.log(this.dataX[1]);

var max = 0;

var min = this.dataY[0];

for (var i = 0; i < this.dataY.length; i++) {

max = Math.max(this.dataY[i],max);

min = Math.min(this.dataY[i],min);

};

console.log(max,min);

//创建一个画布

var canvasChartLine = document.createElement("canvas");

canvasChartLine.width = 400;

canvasChartLine.height = 400;

var canWidth = canvasChartLine.width;

var canHeight = canvasChartLine.height;

var as = ele.getElementsByTagName('canvas');

if (as.length>=1) {

as[0].remove();

};

ele.appendChild(canvasChartLine);

var context = canvasChartLine.getContext("2d");

for (var i = 0; i < this.dataX.length; i++) {

this.dataXLength[i] = this.dataX[i].length;

};

drawChartLineXY(context,canWidth,canHeight,this.dataX.length,min,max,this.dataX,this.dataY,this.dataXLength,this.dataArcSize,this.nameX,this.nameY,this.titleName);

}

function drawChartLineXY(ctx,w,h,l,mi,ma,dataX,dataY,dataXLength,size,nx,ny,title){

var originX = w/15+10;

var originY = h/4\*3;

var yEY = h/16\*3;

var xEX = w - originX -10;

var xUnit = (w - originX\*2 - 30)/l;

var yUnit = parseInt(200/(l+1));

var y = parseInt(ma/l);

var r=size;

var titleLength = title.length;

if(/[\u4e00-\u9fa5]/.test(title)){

titleLength = titleLength\*10; //如果标题是中文

}else{

titleLength = titleLength\*5;//如果标题是英文

}

//y/yUnit = dataY/? ?=dataY[i]\*yUnit/y

ctx.save();

ctx.moveTo(originX,originY);

ctx.lineTo(originX,yEY);

ctx.lineTo(originX-4,yEY+8);

ctx.moveTo(originX,yEY);

ctx.lineTo(originX+4,yEY+8);

ctx.moveTo(originX,originY);

ctx.lineTo(xEX,originY);

ctx.lineTo(xEX-8,originY-4);

ctx.moveTo(xEX,originY);

ctx.lineTo(xEX-8,originY+4);

for (var i = 0; i < l; i++) {

ctx.moveTo(originX+xUnit\*(i+1),originY);

ctx.lineTo(originX+xUnit\*(i+1),originY-5);

if(/[\u4e00-\u9fa5]/.test(dataX[i])){

ctx.fillText(dataX[i],originX+xUnit\*(i+1)-0.5\*dataXLength[i]\*12.5,originY+20);

}//如果是中文

else{

ctx.fillText(dataX[i],originX+xUnit\*(i+1)-0.5\*dataXLength[i]\*6,originY+20);

}//如果是英文

};

for(var j = 0; j < l+1 ;j++){

var textY = y\*(j+1);

ctx.moveTo(originX,originY-(j+1)\*yUnit);

ctx.lineTo(originX+5,originY-(j+1)\*yUnit);

ctx.fillText(textY,originX-20,originY-(j+1)\*yUnit+4);

}

ctx.fillText(nx,xEX+10,originY); //x轴名称

ctx.fillText(ny,5,yEY-10); //y轴名称

ctx.font="18px Verdana";

ctx.fillText(title,w/2-1.5\*originX-0.5\*titleLength,yEY-25);

ctx.stroke();

ctx.restore();

ctx.save();

//画点和折线

for (var a = 0; a < l; a++) {

ctx.moveTo(originX+xUnit\*(a+1),originY-dataY[a]\*yUnit/y);

ctx.arc(originX+xUnit\*(a+1),originY-dataY[a]\*yUnit/y,r,Math.PI\*0,Math.PI\*2);

ctx.fillText(dataY[a],originX+xUnit\*(a+1)-5,originY-dataY[a]\*yUnit/y-10);

// console.log(originX+xUnit\*(a+1),originY-dataY[a]\*yUnit/y);

if(a!=l-1){

ctx.moveTo(originX+xUnit\*(a+1),originY-dataY[a]\*yUnit/y);

ctx.lineTo(originX+xUnit\*(a+2),originY-dataY[a+1]\*yUnit/y);

}

};

ctx.stroke();

ctx.restore();

}

//chartPiePart.js扇形实现模块

var chartPiePartObj = function(){

this.ax;

this.ay;

this.r;

this.context;

this.sAngle;

this.eAngle;

this.fillColor;

this.data;

}

chartPiePartObj.prototype.draw=function(){

var context = this.context;

var x = this.ax;

var y = this.ay;

var r = this.r;

var sAngle = this.sAngle;

var eAngle = this.eAngle;

var fillColor = this.fillColor;

var data = this.data;

Shan(x,y,r,sAngle,eAngle,context,fillColor,data);

}

function Shan(x, y, radius, sDeg, eDeg, ctx , fillColor,data) {

// 初始保存

var midDeg = sDeg + (eDeg-sDeg)/2;

ctx.save();

// 位移到目标点

ctx.translate(x, y);

ctx.beginPath();

// 画出圆弧

ctx.arc(0,0,radius,sDeg, eDeg);

// 再次保存以备旋转

ctx.save();

// 旋转至起始角度

ctx.rotate(eDeg);

// 移动到终点，准备连接终点与圆心

ctx.moveTo(radius,0);

// 连接到圆心

ctx.lineTo(0,0);

// 还原

ctx.restore();

// 旋转至起点角度

ctx.rotate(sDeg);

// 从圆心连接到起点

ctx.lineTo(radius,0);

ctx.closePath();

// 还原到最初保存的状态

ctx.restore();

ctx.fillStyle = fillColor;

ctx.fill();

ctx.save();

ctx.font="16px Arial";

ctx.fillStyle = '#000';

ctx.translate(x, y);

ctx.rotate(midDeg);

ctx.fillText(data,radius+10,0);

ctx.restore();

}

//chartRect.js矩形实现模块

var chartRectObj = function(){

this.x;

this.y;

this.canvas;

this.width;

this.height;

this.fillColor;

this.data;

}

chartRectObj.prototype.init = function(){

var context = this.canvas.getContext("2d");

var x = this.x;

var y = this.y;

var width = this.width;

var height = this.height;

var fillColor = this.fillColor;

var data = this.data;

context.save();

context.fillStyle = fillColor;

context.fillRect(x,y,width,height);

context.restore();

context.save();

context.font="12px Arial";

context.fillStyle = '#000';

context.fillText(data,x+3,y-3);

context.restore();

}

//chartPie.js扇形图实现模块

var chartPieObj = function(){

this.chartPieParts = [];

this.canWidth;

this.canHeight;

this.canvas;

this.colors = []; //暂时是10个数据,可以修改

this.r;

this.datas = [];

this.scale;

this.timer;

this.flag;

this.speed;

}

chartPieObj.prototype.init = function(ele) {

//console.log("chartPieObjinit--");

var canWidth = ele.getAttribute("data-width");

var canHeight = ele.getAttribute("data-height");

this.canvas = document.createElement("canvas");

this.canWidth = canWidth;

this.canHeight = canHeight;

this.r = canWidth/4+25;

this.colors = ["#FF7256","#B9D3EE","#6495ED","#8968CD","#EEE685","#EE9572","#F08080","#388E8E","#68228B","#BC8F8F"];

this.datas = ele.getAttribute("data-chartData").split(",");

this.scale = 0;

this.timer = 0;

this.flag = 0;

this.speed = 50;

this.canvas.width = this.canWidth;

this.canvas.height = this.canHeight;

//console.log("sum",sum);

var as = ele.getElementsByTagName('canvas');

if (as.length>=1) {

as[0].remove();

};

ele.appendChild(this.canvas);//将画布放入网页中

}

chartPieObj.prototype.draw=function(){

var canvas = this.canvas;

var context = canvas.getContext("2d");

var length = this.datas.length;

var sAngle = [];

var eAngle = [];

var sum = datasSum(this.datas);

this.timer = this.timer + lakeDeltatime;

if(this.timer > this.speed){

//console.log(this.timer);

this.timer %= this.speed;

this.scale = Math.min(this.scale + 0.06,1);

}

var scale = this.scale;

context.clearRect(0,0,this.canWidth,this.canHeight);

for(var i=0 ; i<length ; i++){

if(i == 0){

sAngle[i] = 0;

eAngle[i] = (this.datas[i]/sum)\*Math.PI\*2\*scale;

}else{

sAngle[i] = eAngle[i-1];

eAngle[i] = sAngle[i] + (this.datas[i]/sum)\*Math.PI\*2\*scale;

}//设置每一个扇形的起始和结束的度数，such as : Math.PI

this.chartPieParts[i] = new chartPiePartObj();

this.chartPieParts[i].context = context;

this.chartPieParts[i].ax = this.canWidth/2;

this.chartPieParts[i].ay = this.canHeight/2;

this.chartPieParts[i].r = this.r;

this.chartPieParts[i].sAngle = sAngle[i];

this.chartPieParts[i].eAngle = eAngle[i];

this.chartPieParts[i].fillColor = this.colors[i];

this.chartPieParts[i].data = this.datas[i];

this.chartPieParts[i].draw();

}//绘制每个部分的扇形

}

function datasSum(array){

var l = array.length;

var sum = 0;

//console.log(l);

for(var j = 0;j < l;j++){

sum = sum + parseFloat(array[j]);

}

return sum;

}

//chartRingPart.js弧形实现模块

var chartRingPartObj = function(){

this.ax;

this.ay;

this.r;

this.context;

this.sAngle;

this.eAngle;

this.fillColor;

this.lineWidth;

this.data;

}

chartRingPartObj.prototype.draw=function(){

var context = this.context;

var x = this.ax;

var y = this.ay;

var r = this.r;

var sAngle = this.sAngle;

var eAngle = this.eAngle;

var fillColor = this.fillColor;

var lineWidth = this.lineWidth;

var data = this.data;

RingPart(x,y,r,sAngle,eAngle,context,fillColor,lineWidth,data);

}

function RingPart(x,y,radius,sDeg,eDeg,ctx,fillColor,lw,data) {

var midDeg = sDeg + (eDeg-sDeg)/2;

ctx.save();

ctx.lineWidth = lw;

ctx.strokeStyle = fillColor;

ctx.beginPath();

ctx.arc(x,y,radius,sDeg,eDeg);

ctx.stroke();

ctx.restore();

ctx.save();

ctx.font="16px Arial";

ctx.fillStyle = '#000';

ctx.translate(x, y);

ctx.rotate(midDeg);

ctx.fillText(data,radius+30,0);

ctx.restore();

}

//chartRing.js环状图实现模块

var chartRingObj = function(){

this.chartRingParts = [];

this.canWidth;

this.canHeight;

this.canvas;

this.colors = []; //暂时是10个数据,可以修改

this.r;

this.datas = [];

this.scale;

this.timer;

this.flag;

this.speed;

}

chartRingObj.prototype.init = function(ele) {

//console.log("chartPieObjinit--");

var canWidth = ele.getAttribute("data-width");

var canHeight = ele.getAttribute("data-height");

this.canvas = document.createElement("canvas");

this.canWidth = canWidth;

this.canHeight = canHeight;

this.r = canWidth/4;

this.colors = ["#FF7256","#B9D3EE","#6495ED","#8968CD","#EEE685","#EE9572","#F08080","#388E8E","#68228B","#BC8F8F"];

this.datas = ele.getAttribute("data-chartData").split(",");

this.scale = 0;

this.timer = 0;

this.flag = 0;

this.speed = 50;

this.canvas.width = this.canWidth;

this.canvas.height = this.canHeight;

//console.log("sum",sum);

var as = ele.getElementsByTagName('canvas');

if (as.length>=1) {

as[0].remove();

};

ele.appendChild(this.canvas);//将画布放入网页中

}

chartRingObj.prototype.draw=function(){

var canvas = this.canvas;

var context = canvas.getContext("2d");

var length = this.datas.length;

var sAngle = [];

var eAngle = [];

var sum = datasSum(this.datas);

this.timer = this.timer + lakeDeltatime;

if(this.timer > this.speed){

//console.log(this.timer);

this.timer %= this.speed;

this.scale = Math.min(this.scale + 0.06,1);

}

var scale = this.scale;

context.clearRect(0,0,this.canWidth,this.canHeight);

for(var i=0 ; i<length ; i++){

if(i == 0){

sAngle[i] = 0;

eAngle[i] = (this.datas[i]/sum)\*Math.PI\*2\*scale;

}else{

sAngle[i] = eAngle[i-1];

eAngle[i] = sAngle[i] + (this.datas[i]/sum)\*Math.PI\*2\*scale;

}//设置每一个扇形的起始和结束的度数，such as : Math.PI

this.chartRingParts[i] = new chartRingPartObj();

this.chartRingParts[i].context = context;

this.chartRingParts[i].ax = this.canWidth/2;

this.chartRingParts[i].ay = this.canHeight/2;

this.chartRingParts[i].r = this.r;

this.chartRingParts[i].sAngle = sAngle[i];

this.chartRingParts[i].eAngle = eAngle[i];

this.chartRingParts[i].fillColor = this.colors[i];

this.chartRingParts[i].lineWidth = 50;

this.chartRingParts[i].data = this.datas[i];

this.chartRingParts[i].draw();

}//绘制每个部分的扇形

}

function datasSum(array){

var l = array.length;

var sum = 0;

for(var j = 0;j < l;j++){

sum = sum + parseFloat(array[j]);

}

return sum;

}

//chartHistogram.js柱状图实现模块

//这是一个柱状图

var chartHistogramObj = function() {

this.data = [];

this.dataX=[];

this.dataY=[];

this.canWidth;

this.canHeight;

this.canvas;

this.coordinate;

this.scale;

this.colors;

this.timer;

this.dYA;

}

chartHistogramObj.prototype.init = function(ele){

ele.innerHtml = "";

var dXS = ele.getAttribute('data-chartDataX');

var dYS = ele.getAttribute('data-chartDataY');

var dXA = dXS.split(',');

this.dYA = dYS.split(',');

for (var a = 0; a < dXA.length; a++) {

var dataEle = {

"unit": dXA[a],

"data": this.dYA[a]

};

this.data.push(dataEle);

};

var obj = [];

for (var i = 0; i < this.data.length; i++) {

obj[i] = this.data[i];

this.dataX[i] = obj[i].unit;

if(obj[i].data == null) {

obj[i].data = obj[i-1].data;

}

this.dataY[i] = obj[i].data.split(' ');

console.log(obj[i].data.split(' ')[0]);

};

var canWidth = ele.getAttribute("data-width");

var canHeight = ele.getAttribute("data-height");

this.canWidth = canWidth;

this.canHeight = canHeight;

this.scale = 0;

this.colors = ["#FF7256","#B9D3EE","#6495ED","#8968CD","#EEE685","#EE9572","#F08080","#388E8E","#68228B","#BC8F8F"];

var canvas = document.createElement("canvas");

this.canvas = canvas;

canvas.width = this.canWidth;

canvas.height = this.canHeight;

var as = ele.getElementsByTagName('canvas');

if (as.length>=1) {

as[0].remove();

};

ele.appendChild(canvas);

this.coordinate = new chartCoordinateObj();

this.coordinate.canvas = canvas;

this.coordinate.dataX = this.dataX;

this.coordinate.dataY = this.dataY;

this.coordinate.init(); //初始化一个坐标轴

}

chartHistogramObj.prototype.draw = function(){

this.scale = Math.min(this.scale + 0.04,1);

var dataQuantity = this.dataY[0].length;

var rectsWidth = this.coordinate.unitPxX - 10;

var rectWidth = rectsWidth/dataQuantity;//每个柱子的宽度

var max = histogramDataMax(this.dataY);//求dataY中的最大数值

var frag = max/(this.coordinate.yWidth)//一个像素代表多大的值;

var rectHeight = [];

rectHeight = histogramRectHeight(this.dataY,frag); //每个柱子的高du

var rectX = [];

rectX = histogramRectX(rectWidth,this.coordinate.lineX,this.dataY); //每个柱子矩形的x坐标

var canvas = this.canvas;

var context = canvas.getContext("2d");

var originY = this.coordinate.originY;

var chartRects = [];

var colors = this.colors;

context.clearRect(0,0,this.canWidth,this.canHeight);

this.coordinate = new chartCoordinateObj();

this.coordinate.canvas = canvas;

this.coordinate.dataX = this.dataX;

this.coordinate.dataY = this.dataY;

this.coordinate.init(); //初始化一个坐标轴

for (var i = 0; i < this.dataY.length; i++) {

chartRects[i] = [];

for (var j = 0; j < this.dataY[i].length; j++) {

//draw(context,rectX[i][j],rectHeight[i][j],originY,rectWidth,rectHeight[i][j],this.dataY[i][j],this.colors[j]);

chartRects[i][j] = new chartRectObj();

chartRects[i][j].x = rectX[i][j];

chartRects[i][j].y = originY - rectHeight[i][j];

chartRects[i][j].canvas = canvas;

chartRects[i][j].width = rectWidth;

chartRects[i][j].height = rectHeight[i][j]\*this.scale;

chartRects[i][j].fillColor = colors[j];

chartRects[i][j].data = this.dataY[i][j];

chartRects[i][j].init();

// drawDate(canvas,this.dataY[i][j],rectX[i][j],originY - rectHeight[i][j]);

};

};

}

function histogramRectX(rw,lx,dy){

var a = dy.length; //6

var b = dy[0].length; //2

var rectx = [];

for (var i = 0; i < a; i++) {

rectx[i] = [];

for (var j = 0; j < b; j++) {

rectx[i][j] = lx[i] - 5 - rw\*(b-j);

};

};

return rectx;

}

function histogramDataMax(array){

var a = array.length;

var b = array[0].length;

var max = 0;

for (var i = 0; i < a; i++) {

for (var j = 0; j < b; j++) {

max = Math.max(array[i][j],max);

};

};

return max;

}

function histogramRectHeight(array,f){

var a = array.length;

var b = array[0].length;

var heights = [];

for (var i = 0; i < a; i++) {

heights[i] = [];

for (var j = 0; j < b; j++) {

heights[i][j] = array[i][j]/f;

};

};

return heights;

}

―――Lake.js前端开发工具包部分源程序到此全部结束―――

―――接下来是Lake.js前端开发辅助系统源程序―――

//后台入口文件 app.js

//入口文件

var express = require('express');

var morgan = require('morgan');

var path = require('path');

var bodyParser = require('body-parser');

var mongoose = require('mongoose');

var session = require('express-session');

var mongoStore = require('connect-mongo')(session);

var cookieSession = require('cookie-session');//?

var port = 3000 || process.env.PORT; //端口号，可能是在环境变量中的port值或者3000

var app = express();

var dbUrl = "mongodb://localhost/lakeChart";

mongoose.connect(dbUrl);

app.set('views','./app/views/pages'); //设置视图的目录路径

app.set('view engine','jade'); //设置试图的模板引擎

app.use(bodyParser.urlencoded({limit:'50mb',extended:true}));

app.use(bodyParser.json({limit:'50mb'}));

app.use(express.static(path.join(\_\_dirname,'public')));

app.use(session({

secret:'lakeChart',

store:new mongoStore({

url: dbUrl,

collection: 'sessions'

})

}));

require('./config/routes')(app); //路由

app.locals.moment = require('moment');

app.listen(port);

console.log("lakeChart started on " + port);//输出服务是否启动

if('development' === app.get('env')){

app.set('showStackError',true);

app.use(morgan(':method :url :status'));

app.locals.pretty = true;

mongoose.set('debug',true);

}

//routes.js路由部分

var Index = require('../app/controllers/index');

var User = require('../app/controllers/user');

var Layout = require('../app/controllers/layout');

var Layouts = require('../app/controllers/layouts');

var Group = require('../app/controllers/group');

var UserImage = require('../app/controllers/image');

var GroupUser = require('../app/controllers/groupUser');

var \_ = require('underscore');

module.exports = function(app){

app.use(function(req,res,next){

var \_user = req.session.user;

app.locals.user = \_user;

return next();

})

//index page

app.get('/',User.signinRequired,Index.index);

app.get('/signin',User.load);

app.get('/userCenter',User.signinRequired,User.index);

app.post('/user/signin',User.signin);

app.post('/user/signup',User.signup);

app.get('/logout',User.logout);

app.post('/uploadImg',User.uploadImg,User.index);

app.post('/user/setInfo',User.setInfo)

app.get('/createLayout',User.signinRequired,Layout.index);

app.get('/createLayout/:id',Layout.index);

app.post('/saveLayout',Layout.save);

app.post('/updateLayout',Layout.update);

app.post('/queryLayouts',Layouts.query);

app.get('/layout/:id',Layout.query);

app.get('/showLayout/:id',Layout.show);

app.delete('/layoutDelete',Layout.del);

app.post('/uploadImage',Layout.uploadDivBg);

app.post('/queryUserImage',UserImage.query);

app.delete('/deleteImage',UserImage.delImg);

app.post('/createGroup',Group.save);

app.get('/queryGroup',Group.query);

app.post('/deleteGroup',Group.delete);

app.post('/queryGroupers',GroupUser.query);

app.post('/saveGrouper',GroupUser.save);

app.post('/queryUser',GroupUser.queryUser);

}

//user.js有关用户功能部分

var User = require('../models/user');

var fs = require('fs');

var formidable = require('formidable');

var \_ = require('underscore');

exports.index = function(req,res){

console.log('进入用户个人中心----');

var \_user = req.session.user;

console.log('session' + \_user.password);

User.findOne({name:\_user.name},function(err,user){

console.log('-----------nickName' + user.nickName);

res.render('userIndex',{

title: '用户个人中心',

user: user

})

})

}

exports.load = function(req,res){

console.log('进入用户登录页面');

res.render('signin',{

title: '登录页面'

})

}

exports.signup = function(req,res){

console.log('signup start ----')

var \_user = req.body.user;

var userObj = new User(\_user);

User.find({name:\_user.name},function(err,user){

console.log("User find start");

if(err){

console.log(err);

}

if(user.length >= 1){

console.log(\_user.name + "注册失败");

return res.redirect('/');

}else{

userObj.save(function(err,user){

if(err){

console.log(err);

}

})

console.log(\_user.name+"你好,注册成功");

return res.redirect('/signin');

}

})

}

exports.signin = function(req,res){

var \_user = req.body.user;

var name = \_user.name;

var password = \_user.password;

User.findOne({name:name},function(err,user){

if(user){

if(user.password == password){

console.log("登录成功！");

req.session.user = user;

console.log(req.session.user);

}else{

console.log("登录失败！")

}

}else{

console.log("登录失败！");

}

return res.redirect('/');

})

}

exports.logout = function(req,res){

delete req.session.user;

//delete app.locals.user;

res.redirect('/');

}

// midware for user

exports.signinRequired = function(req, res, next) {

var user = req.session.user;

if (!user) {

return res.redirect('/signin');

}

next();

}

exports.setInfo = function(req,res){

console.log("setInfo start --");

var usersess = req.session.user;

var userObj = req.body.user;

var \_user;

console.log(userObj);

User.findOne({name:usersess.name},function(err,user){

\_user = \_.extend(user,userObj);

\_user.save(function(err,user){

res.redirect('/userCenter');

})

})

}

//上传头像并保存到数据库中

exports.uploadImg = function(req,res){

console.log("upload start -- ")

var form = new formidable.IncomingForm();

var userObj = req.session.user;

var \_user;

form.uploadDir = "./public/upload/temp/";

console.log("--uploadDir")

form.parse(req,function(error,fields,files){

for(var key in files){

var file = files[key];

var fName = (new Date()).getTime();

switch(file.type){

case "image/jpeg":

fName = fName + ".jpg";

break;

case "image/png":

fName = fName + ".png";

break;

default:

fName = fName + ".png";

break;

}

console.log(file.size);

var uploadDir = "./public/upload/sculpture/"+fName;

var \_path = "/upload/sculpture/"+fName;

fs.rename(file.path,uploadDir,function(err){

if (err) {

res.write(err+'\n');

res.end();

}

console.log("fName:" + fName);

res.json({path: \_path});

})

userObj.sculpture = "/upload/sculpture/" + fName;

User.findOne({name:userObj.name},function(err,user){

if(user){

\_user = \_.extend(user,userObj);

\_user.save(function(err,user2){

console.log("\_user save DB success!" + user2);

})

}

})

}

})

}

//layouts.js有关布局集功能部分

var User = require('../models/user');

var Layout = require('../models/layout');

var Layouts = require('../models/layouts');

var \_ = require('underscore');

exports.query=function(req,res){

var user = req.session.user;

var userId = user.\_id;

Layouts.findOne({designer:userId},function(err,lays){

if(lays){

//res.json({layouts:lays.layouts});

var array = lays.layouts;

var l = array.length;

var layoutName = [];

var layoutImages = [];

for(var i=0;i<l;i++){

if(array[i]){

Layout.findById(array[i],function(err,lay){

if(err){

console.log(err);

}else{

var layName = lay.name;

var layImage = lay.image;

layoutName.push(layName);

layoutImages.push(layImage);

if(l == layoutName.length){

console.log("querry Success array--"+array.length);

console.log("auerry success"+layImage);

res.json({layouts:lays.layouts,names:layoutName,images:layoutImages});

}

}

})

}

}

}

})

}

//layout.js有关布局功能部分

var User = require('../models/user');

var Layout = require('../models/layout');

var Layouts = require('../models/layouts');

var UserImage = require('../models/image');

var formidable = require('formidable');

var \_ = require('underscore');

var fs = require('fs');

exports.index = function(req,res){

console.log('进入创建主页面--');

if (req.params.id) {

var layId = req.params.id;

Layout.findById({\_id:layId},function(err,lay){

res.render('createIndex',{

title:'显示一个布局',

id:layId,

image: lay.image,

divs:JSON.stringify(lay.layout)

})

})

}else{

res.render('createIndex',{

title:'创建一个布局'

})

}

}

exports.update = function(req,res){

var array = req.body.layout;

var id = req.body.id;

var fName = req.body.fName;

var imgData = req.body.imageData;

var base64Data = imgData.replace(/^data:image\/\w+;base64,/,"");

var dataBuffer = new Buffer(base64Data,'base64');

var userObj = req.session.user;

var userId = userObj.\_id;

var layoutObj = {

'layout':array,

}

var \_lay;

fs.writeFile("./public/upload/sculpture/"+fName+".png",dataBuffer,function(err){

if (err) {

console.log(err);

}else{

Layout.findById({\_id:id},function(err,lay){

\_lay = \_.extend(lay,layoutObj);

\_lay.save(function(err,lay2){

res.json({'success':1});

})

})

}

})

}

//创建一个布局

exports.save = function(req,res){

var array = req.body.layout; //这是一个数组

console.log("save start ---layout"+array.length);

var name = req.body.name;

var imgData = req.body.imageData;

var base64Data = imgData.replace(/^data:image\/\w+;base64,/,"");

var dataBuffer = new Buffer(base64Data,'base64');

var userObj = req.session.user;

var userId = userObj.\_id;

var layoutObj = new Layout();

var layoutsObj;

layoutObj.layout = array;

layoutObj.name = name;

var fName = (new Date()).getTime();

layoutObj.image = fName;

// var \_dirname = "./public/";

fs.writeFile("./public/upload/sculpture/"+fName+".png",dataBuffer,function(err){

if (err) {

console.log(err);

}else{

layoutObj.save(function(err,lay){ //保存一个布局

var layId = lay.\_id;

Layouts.findOne({designer: userId},function(err,lays){ //添加到用户的布局集中

var laysTemp = [];

if(lays){

console.log("添加前"+lays.layouts.length);

laysTemp = lays.layouts;

laysTemp.push(layId);

lays.layouts = laysTemp;

lays.save(function(err,lays2){

console.log("添加成功"+laysTemp);

res.json({'success':1,'id':layId,'fName':fName});

})

}else{

layoutsObj = new Layouts();

laysTemp.push(lay);

layoutsObj.designer = userId;

layoutsObj.layouts = laysTemp;

layoutsObj.save(function(err,lays2){

console.log("新建成功");

res.json({'success':1,'id':layId});

})

}

})

})

}

})

}

exports.query=function(req,res){

console.log("layout:id success");

var layId = req.params.id;

Layout.findById({\_id:layId},function(err,lay){

res.redirect("/showLayout/"+layId);

})

}

exports.show = function(req,res){

console.log("showLayout start");

var layId = req.params.id;

Layout.findById({\_id:layId},function(err,lay){

res.render('showLayout',{

title:'显示一个布局',

layout:lay,

divs:JSON.stringify(lay.layout)

})

})

}

exports.del = function(req,res){

console.log("delete");

var id = req.query.id;

var user = req.session.user;

var userId = user.\_id;

var layObj = {};

if(id){

Layouts.findOne({designer:userId},function(err,lays){

var array = lays.layouts;

var array2 = [];

var l = array.length;

console.log("长度：l = "+l);

console.log("id = "+id);

for (var i = 0; i < l; i++) {

console.log("array[i] = "+array[i]);

console.log("array[i]和id是否相等 = =");

if(array[i] != id){

array2.push(array[i]);

}else{

console.log("相等");

}

}

lays.layouts = array2;

lays.save(function(err,lays2){

console.log("success lays updata");

Layout.remove({\_id:id},function(err,layout){

if(err){

console.log(err);

}else{

res.json({success:1});

}

})

console.log("保存成功");

console.log("layouts的值:"+lays.layouts.length);

})

})

}

}

exports.uploadDivBg = function(req,res){

console.log("upload start -- ");

var userObj = req.session.user;

var userId = userObj.\_id;

var form = new formidable.IncomingForm();

form.uploadDir = "./public/upload/temp/";

console.log("--uploadDir");

form.parse(req,function(error,fields,files){

for(var key in files){

var file = files[key];

var fName = (new Date()).getTime();

switch(file.type){

case "image/jpeg":

fName = fName + ".jpg";

break;

case "image/png":

fName = fName + ".png";

break;

default:

fName = fName + ".png";

break;

}

console.log("hello"+file.type);

console.log(file.path);

var uploadDir = "./public/upload/div/"+fName;

var \_path = "/upload/div/"+fName;

fs.rename(file.path,uploadDir,function(err){

if (err) {

res.write(err+'\n');

res.end();

}

var userImage = new UserImage();

userImage.user = userId;

userImage.url = fName;

userImage.name = 'image1';

userImage.save(function(err,uImage){

console.log("fName:" + uImage.url);

res.json({path: \_path});

});

})

}

})

}

//group.js有关工作组功能部分

var Group = require('../models/group');

var GroupUser = require('../models/groupUser');

var fs = require('fs');

var formidable = require('formidable');

var \_ = require('underscore');

exports.save = function(req,res){

var \_user = req.session.user;

var groupName = req.body.name;

var groupSign = req.body.sign;

var group = new Group();

group.creator = \_user.\_id;

group.name = groupName;

group.uniqueSign = groupSign;

console.log('session' + \_user.password);

console.log('session' + groupName);

console.log('session' + groupName);

group.save(function(err,gp){

console.log("save group!");

res.json({'success':1});

})

}

exports.query = function(req,res){

var \_user = req.session.user;

var userId = \_user.\_id;

Group.find({creator:userId},function(err,gps){

if(err){

console.log(err);

}else{

res.render('queryGroup',{

title:'管理工作组',

groups:gps

})

}

})

}

exports.delete = function(req,res) {

var id = req.body.id;

console.log("delete group id:"+id);

Group.remove({\_id:id},function(err,group){

if (err) {

console.log(err);

}else{

GroupUser.remove({group:id},function(err,group){

if (err) {

console.log(err);

}else{

res.json({"success":1});

}

})

}

})

}

//groupUser.js工作组与用户关系功能部分

var Group = require('../models/group');

var User = require('../models/user');

var GroupUser = require('../models/groupUser');

var fs = require('fs');

var formidable = require('formidable');

var \_ = require('underscore');

exports.save = function(req,res){

var groupId = req.body.id;

var grouper = req.body.userName;

var userState;

User.findOne({name:grouper},function(err,user){

if (err) {

console.log(err);

}else{

if(user.length < 1){

console.log("用户不存在");

userState = 0;

res.json({

"success":1,

"userState":userState

})

}else{

console.log("用户存在"+user['sculpture']);

GroupUser.find({member:user.\_id},function(err,gus){

if(err){

console.log(err);

}

if (gus.length >= 1) {

var flag = 0;

for (var i = 0; i < gus.length; i++) {

if(gus[i].group == groupId){

flag = 0;

}else{

flag = 1;

}

}

if (flag == 0) {

userState = 1;

res.json({

"success":1,

"userState":userState

})

}else{

userState = 2;

var \_groupUser = {

"group":groupId,

"member":user.\_id

}

var groupUserObj = new GroupUser(\_groupUser);

groupUserObj.save(function(err,gur){

if(err){

console.log(err);

}else{

console.log(gur);

res.json({

"success":1,

"userState":userState

})

}

});

}

}else{

console.log("用户不在组中，可以邀请");

userState = 2;

var \_groupUser = {

"group":groupId,

"member":user.\_id

}

var groupUserObj = new GroupUser(\_groupUser);

groupUserObj.save(function(err,gur){

if(err){

console.log(err);

}else{

console.log(gur);

res.json({

"success":1,

"userState":userState

})

}

});

}

})

}

}

})

}

exports.queryUser = function(req,res){

var n = req.body.userName;

User.findOne({name:n},function(err,user){

if (err) {

console.log(err);

}else{

if (user == null) {

}else{

console.log(user.sculpture);

res.json({'user':user});

}

}

})

}

exports.query = function(req,res){

var groupId = req.body.id;

GroupUser.find({group:groupId},function(err,groupUsers){

if (err) {

console.log(err);

}else{

console.log("groupUsers为:"+groupUsers[2]);

var results = [];

var a = [];

for (var i = 0; i < groupUsers.length; i++) {

console.log("start====");

results[i] = function(num){

User.findOne({\_id:groupUsers[num].member},function(err,user){

console.log(groupUsers[num].member);

if(err){

console.log(err);

}else{

a.push(user);

if(a.length == groupUsers.length){

console.log("user 查询成功"+a.length);

res.json({'success':1,'gus':a});

}

}

})

}(i);

}

}

})

}

//image.js有关图片功能部分

var User = require('../models/user');

var UserImage = require('../models/image');

exports.query = function(req,res){

var userObj = req.session.user;

var userId = userObj.\_id;

UserImage.find({user:userId},function(err,userImages){

var imageArray = [];

imageArray = userImages;

res.json({images:imageArray});

})

}

exports.delImg = function(req,res){

var id = req.query.id;

if (id) {

UserImage.remove({url:id},function(err,image){

if (err) {

console.log(err);

}else{

res.json({success:1});

}

})

};

}

//createIndex.js创建布局界面并保存布局 js代码部分

$(function(){

var toolContent = document.getElementById('toolContent');

var imageShowClose = document.getElementById('imageShowClose');

var imageShowContent = document.getElementById('imageShowContent');

var imageShow = document.getElementById('imageShow');

var mask = document.getElementById('mask');

var upload = document.getElementById('upload');

var fileInput = document.getElementById('fileInput');

var deside = document.getElementById('deside');

var changeIndex = document.getElementById('changeIndex');

var upperUI = document.getElementById('upperUI');

var lowerUI = document.getElementById('lowerUI');

var toolwidth =document.getElementById('tool').clientWidth;

var tool = $('#tool');

var page = $('#page');

var divnum = 0;

var scalenum = 0;

var m\_panel,m\_ctrl,m\_type;

var moving = 0,m\_start\_x = 0,m\_start\_y = 0,m\_to\_x = 0,m\_to\_y = 0;

var start\_x,start\_y;

var navToolActive = false;

var navExist = false;

var divActive = page;

var dragdiv;

var desideImg;

var data = document.getElementById("data");

var p = data.getElementsByTagName("p")[0];

var pHtml;

var divData = [];

pHtml = p.innerHTML;

if(pHtml.length > 0){

console.log('有值');

$('#save').css('display','block');

var divs = JSON.parse(pHtml);

var pa = document.getElementById('page');

for (var i = 0; i < divs.length; i++) {

var item = divs[i];

showEle(item,pa);

};

var lakeUIs = $('.lakeUI');

for (var j = 0; j < lakeUIs.length; j++) {

editEvent(lakeUIs[j]);

};

saveEvent();

}

imageShowClose.onclick = function(){

imageShow.style.display = 'none';

mask.style.display = 'none';

}

deside.onclick = function(){

mask.style.display = 'none';

imageShow.style.display = 'none';

desideImg = fileInput.innerHTML;

var divClass = divActive.className;

console.log("divClass:"+divClass);

if (/lakeDiv/i.test(divClass)) {

divActive.style.background = 'url(/upload/div/'+desideImg+')';

}else if(/lakeScaleLarge/i.test(divClass)){

divActive.setAttribute('data-image','/upload/div/'+desideImg);

divActive.innerHTML = '';

scaleLargeLoad();

changePosition(divActive);

divCtrlsClick(divActive);

}else if(/lakeSlideView/i.test(divClass)){

var newSpan = document.createElement('span');

newSpan.setAttribute('data-src','/upload/div/'+desideImg);

divActive.appendChild(newSpan);

updateSlideView();

var children = divActive.getElementsByTagName('span');

toolSpansShow(children,'data-src');

}

}

function disResizable(ele){

var ctrls = ele.getElementsByClassName('ui-Resizable-ctrl');

for (var i = 0; i < ctrls.length; i++) {

ctrls[i].remove();

};

}

$("#file1").bind('change',function(){

var array = this.value.split('\\');

var l = array.length;

var end = l-1;

fileInput.innerHTML = array[end];

})

upload.onclick = function(){

console.log('upload start--');

console.log('uploadname'+fileInput.innerHTML);

var data = new FormData();

var files = $("#file1")[0].files;

var fileName = fileInput.innerHTML;

if (files) {

data.append("file",files[0]);

}

$.ajax({

type:'POST',

dataType:'json',

url:'/uploadImage',

data: data,

contentType:false,

processData:false,

success:function(results){

queryAllImage();

}

})

}

upperUI.onclick = function(){

var targetElement = divActive.nextSibling;

insertAfter(divActive,targetElement);

}

lowerUI.onclick = function(){

console.log('lower')

var target = divActive.previousSibling;

console.log(target);

var parent = divActive.parentNode;

parent.insertBefore(divActive,target);

}

$('#addDiv').bind('click',function(){

var div0 = document.createElement('div');

div0.classList.add('lakeDiv');

div0.classList.add('lakeUI');

div0.setAttribute('id','div' + divnum);

div0.style.width = '50px';

div0.style.height = '50px';

div0.style.border = '1px solid grey';

div0.style.position = 'absolute';

div0.style.left = '100px';

div0.style.top = '100px';

div0.style.background = 'transparent';

page.append(div0);

editEvent(div0);

divnum++;

})

$('#addNav').bind('click',function(){

if(!navExist){

var div0 = document.createElement('div');

div0.className = 'lakeNav';

div0.classList.add('lakeUI');

div0.setAttribute('id','nav');

var span = [];

span[0] = document.createElement('span');

span[1] = document.createElement('span');

span[0].innerHTML = "首页";

span[1].innerHTML = "社区";

div0.appendChild(span[0]);

div0.appendChild(span[1]);

Resizable(div0);

div0.onclick = function(){

toolContent.innerHTML = "";

var spans = this.getElementsByTagName('span');

showTool('nav');

toolSpansShow(spans,'nodeValue');

addSpan.onclick = function(){

var addData = document.createElement('span');

var toolSpans = document.getElementById("toolSpans");

var addInput = document.createElement('input');

addInput.setAttribute('type','text');

addInput.className = 'editInput';

addInput.onblur = function(){

console.log('input onblur--');

var newSpan = document.createElement('span');

newSpan.innerHTML = this.value;

var nav = document.getElementById('nav');

nav.appendChild(newSpan);

var spans = nav.getElementsByTagName('span');

toolSpansShow(spans,'nodeValue');

}

addData.className = 'toolSpan';

addData.appendChild(addInput);

toolSpans.appendChild(addData);

}

}

page.append(div0);

navLoad();

navExist = true;

}

lakeUIHover();

})

$('#addSlideView').bind('click',function(){

var div0 = document.createElement('div');

div0.className = 'lakeSlideView';

div0.classList.add('lakeUI');

div0.setAttribute('data-interval','4000');

var span = [];

span[0] = document.createElement('span');

span[1] = document.createElement('span');

span[0].setAttribute('data-src','/image/slideView1.jpg');

span[1].setAttribute('data-src','/image/slideView2.jpg');

span[0].innerHTML = '';

span[1].innerHTML = '';

div0.appendChild(span[0]);

div0.appendChild(span[1]);

page.append(div0);

slideViewLoad();

div0.style.width = '500px';

div0.style.height = '300px';

div0.style.left = '50px';

div0.style.top = '200px';

editEvent(div0);

})

$('#addButton').bind('click',function(){

var newButton = document.createElement('div');

newButton.className = 'lakeColorButton';

newButton.classList.add('lakeUI');

newButton.setAttribute('data-size','1');

newButton.innerHTML = '确定';

page.append(newButton);

colorButtonLoad();

editEvent(newButton);

})

$("#addPie").bind('click',function(){

var newChart = document.createElement('div');

newChart.className = 'lakeChartPie';

newChart.classList.add('lakeUI');

newChart.setAttribute('data-chartData','30,90,34,60');

newChart.setAttribute('data-chartName','sony,toshiba,lenove,A8');

newChart.setAttribute('data-width','250');

newChart.setAttribute('data-height','250');

page.append(newChart);

chartPiesLoad();

editEvent(newChart);

clearChartPie();

})

$("#addRing").bind('click',function(){

var newChart = document.createElement('div');

newChart.className = 'lakeChartRing';

newChart.classList.add('lakeUI');

newChart.setAttribute('data-chartData','30,90,34,60');

newChart.setAttribute('data-chartName','sony,toshiba,lenove,A8');

newChart.setAttribute('data-width','250');

newChart.setAttribute('data-height','250');

page.append(newChart);

chartRingsLoad();

editEvent(newChart);

clearChartRing();

})

$('#addScaleLarge').bind('click',function(){

var newDiv = document.createElement('div');

newDiv.className = 'lakeScaleLarge';

newDiv.classList.add('lakeUI');

newDiv.setAttribute('id','scaleLarge'+scalenum);

newDiv.setAttribute('data-image','/image/book.jpg');

page.append(newDiv);

scaleLargeLoad();

editEvent(newDiv);

scalenum++;

})

$('#addHistogram').bind('click',function(){

var newDiv = document.createElement('div');

newDiv.className = 'lakeChartHistogram';

newDiv.classList.add('lakeUI');

newDiv.setAttribute('data-width','400');

newDiv.setAttribute('data-height','400');

newDiv.setAttribute('data-chartDataX','一月,二月,三月,四月,五月,六月');

newDiv.setAttribute('data-chartDataY','20 35,25 28,40 36,60 54,43 28,30 32');

page.append(newDiv);

chartHistogramsLoad();

editEvent(newDiv);

clearChartHistogram();

})

$('#returnNav').bind('click',function(){

disTool("navTool");

var toolContent = document.getElementById("toolContent");

toolContent.innerHTML="";

})

$('#saveLayout').bind('click',function(){

//获得第一层的子元素

console.log("saveLayout click!!");

var uiCtrls = [];

uiCtrls = document.getElementsByClassName("ui-Resizable-ctrl");

var l = uiCtrls.length;

console.log("length:\*\*"+uiCtrls.length);

for (var i = l-1; i >= 0; i--) {

console.log("length:end\*\*"+i);

uiCtrls[i].remove();

};

//对网页截图

html2canvas($('#page'),{

onrendered:function(canvas){

var divArray = $('#page').children();

var layoutName = document.getElementById("layoutName");

var name = layoutName.value;

var array = saveDiv(divArray);

var myImage = canvas.toDataURL("image/jpeg");

// console.log(myImage);

$.ajax({

type: 'POST',

dataType: 'json',

url: '/saveLayout',

data: {"layout":array,"name":name,"imageData":myImage},

success:function(results){

var lakeUIs = $('.lakeUI');

for (var i = 0; i < lakeUIs.length; i++) {

editEvent(lakeUIs[i]);

};

if (results.success === 1) {

$('#save').css('display','block');

$('#save').attr('data-id',results.id);

$('#save').attr('data-image',results.fName);

$('#saveLayoutSuccess').fadeIn();

$('#saveLayoutSuccess').css('bottom','60px');

$('#saveLayoutSuccess').delay(3000).hide(0);

saveEvent();

}else{

//save fail

}

}

})

}

})

})

function editEvent(item){

Resizable(item);

if(/lakeSlideView/i.test(item.className)){

item.setAttribute('id','slideView');

updateCtrls(item);

}else if(/lakeColorButton/i.test(item.className)){

btnCtrlsClick(item);

}else if(/lakeScaleLarge/i.test(item.className)){

divCtrlsClick(item);

}else if(/lakeChartPie/i.test(item.className)){

chartPieCtrlsClick(item);

}else if(/lakeChartRing/i.test(item.className)){

chartRingCtrlsClick(item);

}else if(/lakeChartHistogram/i.test(item.className)){

chartHistogramCtrlsClick(item);

}else if(/lakeDiv/i.test(item.className)){

divCtrlsClick(item);

}

lakeUIHover();

}

function saveEvent(){

$('#save').bind('click',function(){

var uiCtrls = [];

uiCtrls = document.getElementsByClassName("ui-Resizable-ctrl");

var l = uiCtrls.length;

console.log("length:\*\*"+uiCtrls.length);

for (var i = l-1; i >= 0; i--) {

console.log("length:end\*\*"+i);

uiCtrls[i].remove();

};

html2canvas($('#page'),{

onrendered:function(canvas){

var array1 = $('#page').children();

var array2 = saveDiv(array1);

var myImage = canvas.toDataURL("image/jpeg");

var id = $('#save').attr('data-id');

var fName = $('#save').attr('data-image');

$.ajax({

type: 'POST',

dataType: 'json',

url: '/updateLayout',

data:{'id':id,'layout':array2,'fName':fName,'imageData':myImage},

success:function(results){

var lakeUIs = $('.lakeUI');

for (var i = 0; i < lakeUIs.length; i++) {

editEvent(lakeUIs[i]);

};

if(results.success === 1){

$('#updateLayoutSuccess').fadeIn();

$('#updateLayoutSuccess').css('bottom','60px');

$('#updateLayoutSuccess').delay(3000).hide(0);

}else{

//update fail

}

}

})

}

})

})

}

function clearChartHistogram(){

var chartexist =document.getElementsByClassName('lakeChartHistogram');

for (var i = 0; i < chartexist.length; i++) {

chartexist[i].innerHTML = '';

Resizable(chartexist[i]);

chartHistogramCtrlsClick(chartexist[i]);

};

}

function clearChartRing(){

var chartexist = document.getElementsByClassName('lakeChartRing');

for (var i = 0; i < chartexist.length; i++) {

chartexist[i].innerHTML = '';

Resizable(chartexist[i]);

chartRingCtrlsClick(chartexist[i]);

};

}

function clearChartPie(){

var chartexist = document.getElementsByClassName('lakeChartPie');

for (var i = 0; i < chartexist.length; i++) {

chartexist[i].innerHTML = '';

Resizable(chartexist[i]);

chartPieCtrlsClick(chartexist[i]);

};

}

function chartHistogramCtrlsClick(parent){

var ctrls = parent.getElementsByClassName('ui-Resizable-ctrl');

for (var i = 0; i < ctrls.length; i++) {

ctrls[i].onclick = function(){

console.log("test---")

if(!navToolActive){

showTool("nav");

}

divActive = this.parentNode;

toolContent.innerHTML = '';

var toolSpans = document.createElement('div');

toolSpans.setAttribute('id','toolSpans');

toolContent.appendChild(toolSpans);

var titspan = document.createElement('div');

titspan.className = 'toolSpan';

titspan.innerHTML = "标签 -- 数据";

toolSpans.appendChild(titspan);

var span=[];

var x=[];

var y=[];

var xInput = [];

var yInput = [];

x=divActive.getAttribute('data-chartDataX').split(',');

y=divActive.getAttribute('data-chartDataY').split(',');

for (var i = 0; i < x.length; i++) {

span[i]=document.createElement('div');

span[i].className = 'toolSpan';

xInput[i] = document.createElement('input');

yInput[i] = document.createElement('input');

xInput[i].className = 'editInput xInput';

yInput[i].className = 'editInput yInput';

xInput[i].value = x[i];

yInput[i].value = y[i];

xInput[i].type = 'text';

yInput[i].type = 'text';

span[i].appendChild(xInput[i]);

span[i].appendChild(yInput[i]);

toolSpans.appendChild(span[i]);

xInput[i].onblur = function(){

xInputHBlur();

}

yInput[i].onblur = function(){

yInputHBlur();

}

};

var add = document.createElement('div');

add.className = 'toolSpan chartadd';

add.innerHTML = '添加';

toolSpans.appendChild(add);

add.onclick = function(){

var newSpan = document.createElement('div');

newSpan.className = 'toolSpan';

var newXInput = document.createElement('input');

var newYInput = document.createElement('input');

newXInput.className = 'editInput xInput';

newYInput.className = 'editInput yInput';

newYInput.type = 'text';

newXInput.type = 'text';

newSpan.appendChild(newXInput);

newSpan.appendChild(newYInput);

toolSpans.insertBefore(newSpan,this);

newXInput.onblur = function(){

xInputHBlur();

}

newYInput.onblur = function(){

yInputHBlur();

}

}

}

};

}

function chartRingCtrlsClick(parent){

var ctrls = parent.getElementsByClassName('ui-Resizable-ctrl');

for (var i = 0; i < ctrls.length; i++) {

ctrls[i].onclick = function(){

if(!navToolActive){

showTool("nav");

}

divActive = this.parentNode;

toolContent.innerHTML = '';

var toolSpans = document.createElement('div');

toolSpans.setAttribute('id','toolSpans');

toolContent.appendChild(toolSpans);

var titspan = document.createElement('div');

titspan.className = 'toolSpan';

titspan.innerHTML = "标签 -- 数据";

toolSpans.appendChild(titspan);

var span=[];

var x=[];

var y=[];

var xInput = [];

var yInput = [];

x=divActive.getAttribute('data-chartName').split(',');

y=divActive.getAttribute('data-chartData').split(',');

for (var i = 0; i < x.length; i++) {

span[i]=document.createElement('div');

span[i].className = 'toolSpan';

xInput[i] = document.createElement('input');

yInput[i] = document.createElement('input');

xInput[i].className = 'editInput xInput';

yInput[i].className = 'editInput yInput';

xInput[i].value = x[i];

yInput[i].value = y[i];

xInput[i].type = 'text';

yInput[i].type = 'text';

span[i].appendChild(xInput[i]);

span[i].appendChild(yInput[i]);

toolSpans.appendChild(span[i]);

xInput[i].onblur = function(){

xInputRingBlur();

}

yInput[i].onblur = function(){

yInputRingBlur();

}

};

var add = document.createElement('div');

add.className = 'toolSpan chartadd';

add.innerHTML = '添加';

toolSpans.appendChild(add);

add.onclick = function(){

var newSpan = document.createElement('div');

newSpan.className = 'toolSpan';

var newXInput = document.createElement('input');

var newYInput = document.createElement('input');

newXInput.className = 'editInput xInput';

newYInput.className = 'editInput yInput';

newYInput.type = 'text';

newXInput.type = 'text';

newSpan.appendChild(newXInput);

newSpan.appendChild(newYInput);

toolSpans.insertBefore(newSpan,this);

newXInput.onblur = function(){

xInputRingBlur();

}

newYInput.onblur = function(){

yInputRingBlur();

}

}

}

}

}

function chartPieCtrlsClick(parent){

var ctrls = parent.getElementsByClassName('ui-Resizable-ctrl');

for (var i = 0; i < ctrls.length; i++) {

ctrls[i].onclick = function(){

if(!navToolActive){

showTool("nav");

}

divActive = this.parentNode;

toolContent.innerHTML = '';

var toolSpans = document.createElement('div');

toolSpans.setAttribute('id','toolSpans');

toolContent.appendChild(toolSpans);

var titspan = document.createElement('div');

titspan.className = 'toolSpan';

titspan.innerHTML = "标签 -- 数据";

toolSpans.appendChild(titspan);

var span=[];

var x=[];

var y=[];

var xInput = [];

var yInput = [];

x=divActive.getAttribute('data-chartName').split(',');

y=divActive.getAttribute('data-chartData').split(',');

for (var i = 0; i < x.length; i++) {

span[i]=document.createElement('div');

span[i].className = 'toolSpan';

xInput[i] = document.createElement('input');

yInput[i] = document.createElement('input');

xInput[i].className = 'editInput xInput';

yInput[i].className = 'editInput yInput';

xInput[i].value = x[i];

yInput[i].value = y[i];

xInput[i].type = 'text';

yInput[i].type = 'text';

span[i].appendChild(xInput[i]);

span[i].appendChild(yInput[i]);

toolSpans.appendChild(span[i]);

xInput[i].onblur = function(){

xInputPieBlur();

}

yInput[i].onblur = function(){

yInputPieBlur();

}

};

var add = document.createElement('div');

add.className = 'toolSpan chartadd';

add.innerHTML = '添加';

toolSpans.appendChild(add);

add.onclick = function(){

var newSpan = document.createElement('div');

newSpan.className = 'toolSpan';

var newXInput = document.createElement('input');

var newYInput = document.createElement('input');

newXInput.className = 'editInput xInput';

newYInput.className = 'editInput yInput';

newYInput.type = 'text';

newXInput.type = 'text';

newSpan.appendChild(newXInput);

newSpan.appendChild(newYInput);

toolSpans.insertBefore(newSpan,this);

newXInput.onblur = function(){

xInputPieBlur();

}

newYInput.onblur = function(){

yInputPieBlur();

}

}

}

}

}

function xInputPieBlur(){

var ex = toolSpans.getElementsByClassName('editInput xInput');

var exv = [];

for (var i = 0; i < ex.length; i++) {

exv[i] = ex[i].value;

};

var exs = exv.join();

divActive.setAttribute('data-chartName',exs);

divActive.innerHTML = "";

chartPiesLoad();

clearChartPie();

}

function yInputPieBlur(){

var ex = toolSpans.getElementsByClassName('editInput yInput');

var exv = [];

for (var i = 0; i < ex.length; i++) {

exv[i] = ex[i].value;

};

var exs = exv.join();

divActive.setAttribute('data-chartData',exs);

divActive.innerHTML = "";

chartPiesLoad();

clearChartPie();

}

function xInputHBlur(){

var ex = toolSpans.getElementsByClassName('editInput xInput');

var exv = [];

for (var i = 0; i < ex.length; i++) {

exv[i] = ex[i].value;

};

var exs = exv.join();

divActive.setAttribute('data-chartDataX',exs);

divActive.innerHTML = "";

chartHistogramsLoad();

clearChartHistogram();

}

function yInputHBlur(){

var ex = toolSpans.getElementsByClassName('editInput yInput');

var exv = [];

for (var i = 0; i < ex.length; i++) {

exv[i] = ex[i].value.toString();

};

var exs = exv.join();

divActive.setAttribute('data-chartDataY',exs.toString());

divActive.innerHTML = "";

chartHistogramsLoad();

clearChartHistogram();

}

function xInputRingBlur(){

var ex = toolSpans.getElementsByClassName('editInput xInput');

var exv = [];

for (var i = 0; i < ex.length; i++) {

exv[i] = ex[i].value;

};

var exs = exv.join();

divActive.setAttribute('data-chartName',exs);

divActive.innerHTML = "";

chartRingsLoad();

clearChartRing();

}

function yInputRingBlur(){

var ex = toolSpans.getElementsByClassName('editInput yInput');

var exv = [];

for (var i = 0; i < ex.length; i++) {

exv[i] = ex[i].value;

};

var exs = exv.join();

divActive.setAttribute('data-chartData',exs);

divActive.innerHTML = "";

chartRingsLoad();

clearChartRing();

}

function btnCtrlsClick(parent){

var ctrls = parent.getElementsByClassName('ui-Resizable-ctrl');

for (var i = 0; i < ctrls.length; i++) {

ctrls[i].onclick = function(){

if(!navToolActive){

showTool("nav");

}

divActive = this.parentNode;

toolContent.innerHTML = '';

var toolSpans = document.createElement('div');

toolSpans.setAttribute('id','toolSpans');

var span0 = document.createElement('div');

var span1 = document.createElement('div');

var nameInput = document.createElement('input');

nameInput.type = 'text';

nameInput.value = divActive.firstChild.nodeValue;

nameInput.className = 'editInput';

span0.className = 'toolSpan';

span0.innerHTML = "按钮：";

span0.appendChild(nameInput);

//onblur

nameInput.onblur = function(){

divActive.firstChild.nodeValue = this.value;

}

span1.className = 'toolSpan';

var sizeInput = document.createElement('input');

sizeInput.type = 'number';

sizeInput.value = divActive.getAttribute('data-size');

sizeInput.className = 'editInput';

span1.innerHTML = "大小：";

span1.appendChild(sizeInput);

sizeInput.onblur = function(){

divActive.setAttribute('data-size',this.value);

divActive.innerHTML = '';

divActive.innerHTML = nameInput.value;

colorButtonLoad();

changePosition(divActive);

btnCtrlsClick(divActive);

}

toolSpans.appendChild(span0);

toolSpans.appendChild(span1);

toolContent.appendChild(toolSpans);

}

};

}

function queryAllImage(){

$.ajax({

type:'POST',

dataType:'json',

url:'/queryUserImage',

success:function(results){

var userImages = results.images;

imageShowContent.innerHTML = '';

for (var i = 0; i < userImages.length; i++) {

var div = document.createElement('div');

var divh = document.createElement('div');

div.className = 'imgDiv';

divh.className = 'imgDivHover';

var img = document.createElement('img');

img.src = '/upload/div/'+userImages[i].url;

img.setAttribute('id',userImages[i].url);

var icon = document.createElement('i');

icon.className = 'iconfont iconClose';

icon.innerHTML = "&#xf000a;";

divh.appendChild(icon);

div.appendChild(img);

div.appendChild(divh);

icon.addEventListener('click',function(){

var deleteimg = this.parentNode.previousSibling;

deleteImage(deleteimg);

})

img.addEventListener('mouseover',function(){

this.nextSibling.style.display = 'block';

})

img.addEventListener('mouseout',function(){

this.nextSibling.style.display = 'none';

})

divh.addEventListener('mouseover',function(){

this.style.display = 'block';

})

divh.addEventListener('mouseout',function(){

this.style.display = 'none';

})

imageShowContent.appendChild(div);

img.onclick = function(){

fileInput.innerHTML = this.id;

}

};

}

})

}

function deleteImage(img){

var id = img.getAttribute('id');

console.log(id);

$.ajax({

type:'DELETE',

url:"/deleteImage?id="+id

}).done(function(results){

if (results.success) {

queryAllImage();

};

})

}

function toolSpansShow(spans,type){

console.log('toolSpansShow start - -');

toolContent.innerHTML = "";

var toolSpans = document.createElement('div');

toolSpans.setAttribute('id','toolSpans');

var addSpan = document.createElement('div');

addSpan.setAttribute('id','addSpan');

addSpan.innerHTML = "添加";

if(type == 'nodeValue'){

console.log('addSpan nav\*\*');

addSpan.onclick = function(){

console.log('click addSpan');

var addData = document.createElement('span');

var toolSpans = document.getElementById("toolSpans");

var addInput = document.createElement('input');

addInput.setAttribute('type','text');

addInput.style.color = 'black';

addInput.style.height = '50px';

addInput.style.width = '100px';

addInput.style.paddingLeft = "20px";

//input 失去焦点时 将value处理

addInput.onblur = function(){

console.log('input onBlur--');

var newSpan = document.createElement('span');

newSpan.innerHTML = this.value;

var nav = document.getElementById('nav');

nav.appendChild(newSpan);

var spans = nav.getElementsByTagName('span');

toolSpansShow(spans,'nodeValue');

}

addData.className = 'toolSpan';

addData.appendChild(addInput);

toolSpans.appendChild(addData);

}

}else if(type == 'data-src'){

addSpan.onclick = function(){

//获取所有图片，选择图片

console.log('assSpan');

mask.style.display = 'block';

imageShow.style.display = 'block';

queryAllImage();

}

}

toolContent.appendChild(toolSpans);

toolContent.appendChild(addSpan);

for (var i = 0; i < spans.length; i++) {

var array = [];

var iconClose=[];

array[i] = document.createElement('span');

iconClose[i] = document.createElement('i');

iconClose[i].className = 'iconfont iconClose';

iconClose[i].setAttribute('id','iconClose'+i);

iconClose[i].innerHTML= "&#xf000a;";

iconClose[i].onclick = function(){

var iconCloseId = this.getAttribute('id');

var id = iconCloseId.split('iconClose')[1];

var toolSpans = document.getElementById("toolSpans");

var is = toolSpans.getElementsByTagName('i');

toolSpans.getElementsByTagName('span')[id].remove();

if(type == 'nodeValue'){

var nav = divActive;

nav.getElementsByTagName('span')[id].remove();

}else if(type == 'data-src'){

var slideView = divActive;

var slideViewSpan = slideView.getElementsByTagName('span');

slideViewSpan[id].remove();

updateSlideView();

}

for (var i = 0; i < is.length; i++) {

is[i].setAttribute('id','iconClose'+i);

};

}

if(type == 'nodeValue'){

array[i].innerHTML = spans[i].firstChild.nodeValue;

}else if(type == 'data-src'){

array[i].innerHTML = "<img src='"+spans[i].getAttribute('data-src')+"'>";

}

array[i].className = 'toolSpan';

array[i].appendChild(iconClose[i]);

toolSpans.appendChild(array[i]);

};

}

function updateSlideView(){

slideViewLoad();

// disResizable(slideView);

// Resizable(slideView);

editEvent(divActive);

updateCtrls(divActive);

}

function divCtrlsClick(parent){

var ctrls = parent.getElementsByClassName('ui-Resizable-ctrl');

for (var i = 0; i < ctrls.length; i++) {

ctrls[i].onclick = function(){

if(!navToolActive){

showTool("nav");

}

divActive = this.parentNode;

// var id = this.parentNode.getAttribute('id');

var width = this.parentNode.offsetWidth;

var height = this.parentNode.offsetHeight;

var left = this.parentNode.style.left;

var top = this.parentNode.style.top;

var bg = this.parentNode.style.background;

toolContent.innerHTML = "";

var toolSpans = document.createElement('div');

toolSpans.setAttribute('id','toolSpans');

toolContent.appendChild(toolSpans);

var widthSpan = document.createElement('span');

widthSpan.innerHTML = "宽度："+width;

widthSpan.className = 'toolSpan';

var heightSpan = document.createElement('span');

heightSpan.innerHTML = "高度："+height;

heightSpan.className = 'toolSpan';

var leftSpan = document.createElement('span');

leftSpan.innerHTML = "左边距："+left;

leftSpan.className = 'toolSpan';

var topSpan = document.createElement('span');

topSpan.innerHTML = "上边距："+top;

topSpan.className = 'toolSpan';

var bgSpan = document.createElement('span');

bgSpan.innerHTML = "背景：<span id='bgEdit'><i class='icon iconfont'>&#xe729;</i></span>";

bgSpan.className = 'toolSpan';

toolSpans.appendChild(widthSpan);

toolSpans.appendChild(heightSpan);

toolSpans.appendChild(leftSpan);

toolSpans.appendChild(topSpan);

toolSpans.appendChild(bgSpan);

var bgEdit = document.getElementById('bgEdit');

bgEdit.onclick = function(){

mask.style.display = 'block';

imageShow.style.display = 'block';

//获取当前用户上传的所有的图片

queryAllImage();

}

}

};

}

function updateCtrls(parent){

var ctrls = parent.getElementsByClassName('ui-Resizable-ctrl');

for (var i = 0; i < ctrls.length; i++) {

ctrls[i].onclick = function(){

if(!navToolActive){

showTool("nav");

}

divActive = this.parentNode;

var children = divActive.getElementsByTagName('span');

toolSpansShow(children,'data-src');

addSpan.onclick = function(){

//获取所有图片，选择图片

mask.style.display = 'block';

imageShow.style.display = 'block';

queryAllImage();

}

}

};

}

function showTool(name){

var toolName = name+"Tool";

var tool = document.getElementById(toolName);

tool.style.left = "0";

navToolActive = true;

}

function disTool(name){

var tool = document.getElementById(name);

tool.style.left = "-20%";

navToolActive = false;

}

function saveDiv(divArray){

var l = divArray.length;

var array = [];

for(var i=0;i<l;i++){

var eleObj = objNode(divArray[i]);

array.push(eleObj);

}

return array;

}

// 返回一个对象

function objNode(node){

var width = node.style.width;

var height = node.style.height;

var left = node.style.left;

var top = node.style.top;

var className = node.className;

var tagName = node.tagName;

var childs = [];

var bg = node.style.background;

var data;

var text = "";

if(/lakeSlideView/i.test(className)){

data = node.getAttribute('data-interval');

}else if(/lakeColorButton/i.test(className)){

console.log("lake color button save");

data = node.getAttribute('data-size');

text = node.firstChild.nodeValue;

}else if(/lakeScaleLarge/i.test(className)){

data = node.getAttribute('data-image');

}else if(/lakeChartPie/i.test(className)){

data = {

'dataX':node.getAttribute('data-chartName'),

'dataY':node.getAttribute('data-chartData'),

'cWidth':node.getAttribute('data-width'),

'cHeight':node.getAttribute('data-height')

}

}else if(/lakeChartRing/i.test(className)){

data = {

'dataX':node.getAttribute('data-chartName'),

'dataY':node.getAttribute('data-chartData'),

'cWidth':node.getAttribute('data-width'),

'cHeight':node.getAttribute('data-height')

}

}else if(/lakeChartHistogram/i.test(className)){

data = {

"dataX":node.getAttribute('data-chartDataX'),

"dataY":node.getAttribute('data-chartDataY'),

"cWidth":node.getAttribute('data-width'),

"cHeight":node.getAttribute('data-height')

}

}

var state;

var childEles = node.getElementsByTagName('div');

var childEle = [];

for (var i = 0; i < childEles.length; i++) {

if(childEles[i].parentNode == node){

childEle.push(childEles[i]);

}

};

var childSpans = node.getElementsByTagName('span');

var childSpan = [];

for (var i = 0; i < childSpans.length; i++) {

if(childSpans[i].parentNode == node){

childSpan.push(childSpans[i]);

}

};

console.log('childSpan.length:'+childSpan.length);

console.log('childEle.length:'+childEle.length);

if(childSpan.length > 0&&childEle.length == 0){

state = "spanmarry";

console.log('有span子元素');

for (var j = 0; j < childSpan.length; j++) {

console.log(childSpan[j].tagName);

var spanObj = {

"type": "span",

"text": childSpan[j].innerHTML,

"marginLeft":childSpan[j].style.marginLeft,

"marginTop": childSpan[j].style.marginTop,

"className": childSpan[j].className,

"data-src": childSpan[j].getAttribute("data-src")

}

childs.push(spanObj);

};

}else if(childEle.length > 0&&childSpan.length == 0){

state = "divmarry";

console.log('有div子元素');

for (var i = 0; i < childEle.length; i++) {

var divObj = objNode(childEle[i]);

childs.push(divObj);

};

}else if(childEle.length > 0 && childSpan.length > 0){

state = "marry";

console.log("既有span又有div");

for (var a = 0; a < childSpan.length; a++) {

console.log(childSpan[a].tagName);

var spanObj = {

"tagName": "span",

"text": childSpan[a].innerHTML,

"marginLeft":childSpan[a].style.marginLeft,

"marginTop": childSpan[a].style.marginTop,

"className": childSpan[a].className,

"data-src": childSpan[a].getAttribute("data-src")

}

childs.push(spanObj);

};

for (var b = 0; b < childEle.length; b++) {

var divObj = objNode(childEle[b]);

childs.push(divObj);

};

}else{

state = "single";

}

//对象化的元素

var eleObj;

eleObj = {

"width": width,

"height": height,

"left": left,

"top": top,

"className":className,

"childs":childs,

"background": bg,

"tagName":tagName,

"state":state,

"data":data,

"text":text

}

return eleObj;

}

function changePosition(e){

var panel = e;

panel.setAttribute('draggable',true);

panel.classList.add('draggable');

var t=document.createElement("div");

t.class=t.className="ui-Resizable-t ui-Resizable-ctrl";

panel.appendChild(t);

t.addEventListener('mousedown',function(e){

mouseDown(e,panel,t,'t');

})

var boxes = document.querySelectorAll('.draggable');

[].forEach.call(boxes,function(box){

box.addEventListener('dragstart',handleDragStart,false);

box.addEventListener('dragenter',handleDragEnter,false);

box.addEventListener('dragover',handleDragOver,false);

box.addEventListener('dragleave',handleDragLeave,false);

box.addEventListener('drag',handleDrag,false);

box.addEventListener('drop',handleDrop,false);

box.addEventListener('dragend',handleDragEnd,false);

})

}

function Resizable(p){

var panel=p;

p.setAttribute('draggable',true);

var r=document.createElement("div");

var b=document.createElement("div");

var rb=document.createElement("div");

var t=document.createElement("div");

t.class=t.className="ui-Resizable-t ui-Resizable-ctrl";

r.class=r.className="ui-Resizable-r ui-Resizable-ctrl";

b.class=b.className="ui-Resizable-b ui-Resizable-ctrl";

rb.class=rb.className="ui-Resizable-rb ui-Resizable-ctrl";

panel.appendChild(t);

panel.appendChild(r);

panel.appendChild(b);

panel.appendChild(rb);

t.addEventListener('mousedown',function(e){

mouseDown(e,panel,t,'t');

})

r.addEventListener('mousedown',function(e){

mouseDown(e,panel,r,'r');

});

b.addEventListener('mousedown',function(e){

mouseDown(e,panel,b,'b');

})

rb.addEventListener('mousedown',function(e){

mouseDown(e,panel,rb,'rb');

})

p.addEventListener('dragstart',handleDragStart,false);

p.addEventListener('dragenter',handleDragEnter,false);

p.addEventListener('dragover',handleDragOver,false);

p.addEventListener('dragleave',handleDragLeave,false);

p.addEventListener('drag',handleDrag,false);

p.addEventListener('drop',handleDrop,false);

p.addEventListener('dragend',handleDragEnd,false);

}

function mouseDown(e,panel,ctrl,type){

var e = e||window.event;

e.preventDefault();

m\_start\_x = e.pageX - ctrl.offsetLeft;

m\_start\_y = e.pageY - ctrl.offsetTop;

start\_x = e.pageX - panel.offsetLeft;

start\_y = e.pageY - panel.offsetTop;

m\_panel = panel;

m\_ctrl = ctrl;

m\_type = type;

moving = setInterval(mouseMove,10);

}

function mouseMove(){

if(moving){

var to\_x = m\_to\_x - m\_start\_x;

var to\_y = m\_to\_y - m\_start\_y;

var min\_x = m\_ctrl.offsetLeft;

var min\_y = m\_panel.offsetTop;

var move\_x = m\_to\_x - start\_x;

var move\_y = m\_to\_y - start\_y;

switch(m\_type){

case 't':

m\_panel.style.left = move\_x + 'px';

m\_panel.style.top = move\_y + 'px';

break;

case 'r':

m\_ctrl.style.left = to\_x + 'px';

//m\_ctrl.style.top = to\_y + 'px';

m\_panel.style.width = to\_x +10+ 'px';

break;

case 'b':

//m\_ctrl.style.left = to\_x + 'px';

m\_ctrl.style.top = to\_y + 'px';

m\_panel.style.height = to\_y +10+ 'px';

break;

case 'rb':

m\_ctrl.style.left = to\_x + 'px';

m\_ctrl.style.top = to\_y + 'px';

m\_panel.style.width = to\_x +20+ 'px';

m\_panel.style.height = to\_y +20+ 'px';

break;

}

}

}

document.onmousemove=function(e){

var e=window.event||e;

m\_to\_x = Math.max(toolwidth,e.pageX);

m\_to\_y = Math.max(0,e.pageY);

}

document.onmouseup = function(){

clearInterval(moving);

moving = 0;

var cls = document.getElementsByClassName('ui-Resizable-ctrl');

for(var i = 0;i < cls.length;i++){

cls[i].style.left = "";

cls[i].style.top = "";

}

}

//以下为拖放的事件函数

function handleDragStart(e){

this.style.opacity = '0.4';

e.dataTransfer.effectAllowed = 'all';

dragdiv = this;

console.log(dragdiv);

}

function handleDragOver(e){

if(e.preventDefault){

e.preventDefault();

}

e.dataTransfer.dropEffect = 'link';

return false;

}

function handleDragEnter(e){

this.classList.add('over');

}

function handleDragLeave(e){

this.classList.remove('over');

}

function handleDrag(e){

//console.log('drag',this);

}

function handleDrop(e){

if(e.stopPorpagation){

e.stopPorpagation();

}

console.log("drop :"+this);

dragdiv.style.left = '5px';

dragdiv.style.top = '5px';

e.target.appendChild(dragdiv);

return false;

}

function handleDragEnd(e){

this.style.opacity = '1';

var boxes = document.querySelectorAll('.draggable');

[].forEach.call(boxes,function(box){

box.classList.remove('over');

});

}

function lakeUIHover(){

var lakeUIs = document.getElementsByClassName('lakeUI');

for (var i = 0; i < lakeUIs.length; i++) {

lakeUIs[i].addEventListener('mouseover',function(){

var ctrls = this.getElementsByClassName('ui-Resizable-ctrl');

for (var j = 0; j < ctrls.length; j++) {

ctrls[j].style.background = 'grey';

ctrls[j].style.opacity = '0.2';

};

var width = this.offsetWidth;

var height = this.offsetHeight;

var wh = document.createElement('div');

wh.setAttribute('id','wh');

wh.style.position = 'absolute';

wh.style.right = '10px';

wh.style.top = '-16px';

wh.style.color = 'grey';

wh.style.fontFamily = 'Microsoft Yahei';

wh.style.fontSize = '10px';

wh.innerHTML = width+'px\*'+height+'px';

this.appendChild(wh);

})

lakeUIs[i].addEventListener('mouseout',function(){

var ctrls = this.getElementsByClassName('ui-Resizable-ctrl');

for (var j = 0; j < ctrls.length; j++) {

ctrls[j].style.background = 'transparent';

};

var wh = document.getElementById('wh');

wh.remove();

})

};

}

function insertAfter(newElement,targetElement){

var parent = targetElement.parentNode;

if(parent.lastChild == targetElement){

parent.appendChild(newElement);

}else{

parent.insertBefore(newElement,targetElement.nextSibling);

}

}

function showEle(item,pa){

var ele = document.createElement(item.tagName);

ele.style.width = item.width;

ele.style.height = item.height;

ele.style.top = item.top;

ele.style.left = item.left;

ele.className = item.className;

ele.style.background = item.background;

if(/lakeSlideView/i.test(item.className)){

ele.setAttribute('data-interval',item['data']);

}else if(/lakeColorButton/i.test(item.className)){

ele.setAttribute('data-size',item['data']);

ele.innerHTML = item.text;

}else if(/lakeScaleLarge/i.test(item.className)){

ele.setAttribute('data-image',item['data']);

}else if(/lakeChartPie/i.test(item.className)){

ele.setAttribute('data-chartName',item['data'].dataX);

ele.setAttribute('data-chartData',item['data'].dataY);

ele.setAttribute('data-width',item['data'].cWidth);

ele.setAttribute('data-height',item['data'].cHeight);

}else if(/lakeChartRing/i.test(item.className)){

ele.setAttribute('data-chartName',item['data'].dataX);

ele.setAttribute('data-chartData',item['data'].dataY);

ele.setAttribute('data-width',item['data'].cWidth);

ele.setAttribute('data-height',item['data'].cHeight);

}else if(/lakeChartHistogram/i.test(item.className)){

ele.setAttribute('data-chartDataX',item['data'].dataX);

ele.setAttribute('data-chartDataY',item['data'].dataY);

ele.setAttribute('data-width',item['data'].cWidth);

ele.setAttribute('data-height',item['data'].cHeight);

}

if(item.state == "spanmarry"){

for (var i = 0; i < item.childs.length; i++) {

var span = document.createElement('span');

span.innerHTML = item.childs[i].text;

if(item.childs[i].marginLeft!=null) span.style.marginLeft = item.childs[i].marginLeft;

if(item.childs[i].marginTop!=null) span.style.marginTop = item.childs[i].marginTop;

if(item.childs[i].className!=null) span.className = item.childs[i].className;

if(item.childs[i]['data-src']!=null) span.setAttribute('data-src',item.childs[i]['data-src']);

ele.appendChild(span);

};

}else if(item.state == "divmarry"){

for (var j = 0; j < item.childs.length; j++) {

showEle(item.childs[j],ele);

};

}else if(item.state == "marry"){

for (var k = 0; k < item.childs.length; k++) {

if(item.childs[k].tagName == "span"){

var span = document.createElement('span');

span.innerHTML = item.childs[k].text;

if(item.childs[k].marginLeft!=null) span.style.marginLeft = item.childs[k].marginLeft;

if(item.childs[k].marginTop!=null) span.style.marginTop = item.childs[k].marginTop;

if(item.childs[k].className!=null) span.className = item.childs[k].className;

if(item.childs[k]['data-src']!=null) span.setAttribute('data-src',item.childs[k]['data-src']);

ele.appendChild(span);

}else{

showEle(item.childs[k],ele);

}

};

}

pa.appendChild(ele);

}

})

//展现布局界面js代码

$(function(){

var page = document.getElementById("page");

var data = document.getElementById("data");

var p = data.getElementsByTagName("p")[0];

var pHtml;

var divData = [];

$('#show').hide();

pHtml = p.firstChild.nodeValue;

var divs = JSON.parse(pHtml);

for (var i = 0; i < divs.length; i++) {

var item = divs[i];

showEle(item,page);

};

$('#output').bind('click',function(){

var pageHtml = page.innerHTML;

pageHtml = ToHtmlString(pageHtml);

$('#show').html(pageHtml);

$('#show').fadeIn();

$('#mask').show();

})

$('#mask').bind('click',function(){

$(this).fadeOut();

$('#show').hide();

})

})

function showEle(item,pa){

var ele = document.createElement(item.tagName);

ele.style.width = item.width;

ele.style.height = item.height;

ele.style.top = item.top;

ele.style.left = item.left;

ele.className = item.className;

ele.style.background = item.background;

if(/lakeSlideView/i.test(item.className)){

ele.setAttribute('data-interval',item['data']);

}else if(/lakeColorButton/i.test(item.className)){

ele.setAttribute('data-size',item['data']);

ele.innerHTML = item.text;

}else if(/lakeScaleLarge/i.test(item.className)){

ele.setAttribute('data-image',item['data']);

}else if(/lakeChartPie/i.test(item.className)){

ele.setAttribute('data-chartName',item['data'].dataX);

ele.setAttribute('data-chartData',item['data'].dataY);

ele.setAttribute('data-width',item['data'].cWidth);

ele.setAttribute('data-height',item['data'].cHeight);

}else if(/lakeChartRing/i.test(item.className)){

ele.setAttribute('data-chartName',item['data'].dataX);

ele.setAttribute('data-chartData',item['data'].dataY);

ele.setAttribute('data-width',item['data'].cWidth);

ele.setAttribute('data-height',item['data'].cHeight);

}else if(/lakeChartHistogram/i.test(item.className)){

ele.setAttribute('data-chartDataX',item['data'].dataX);

ele.setAttribute('data-chartDataY',item['data'].dataY);

ele.setAttribute('data-width',item['data'].cWidth);

ele.setAttribute('data-height',item['data'].cHeight);

}

if(item.state == "spanmarry"){

for (var i = 0; i < item.childs.length; i++) {

var span = document.createElement('span');

span.innerHTML = item.childs[i].text;

if(item.childs[i].marginLeft!=null) span.style.marginLeft = item.childs[i].marginLeft;

if(item.childs[i].marginTop!=null) span.style.marginTop = item.childs[i].marginTop;

if(item.childs[i].className!=null) span.className = item.childs[i].className;

if(item.childs[i]['data-src']!=null) span.setAttribute('data-src',item.childs[i]['data-src']);

ele.appendChild(span);

};

}else if(item.state == "divmarry"){

for (var j = 0; j < item.childs.length; j++) {

showEle(item.childs[j],ele);

};

}else if(item.state == "marry"){

for (var k = 0; k < item.childs.length; k++) {

if(item.childs[k].tagName == "span"){

var span = document.createElement('span');

span.innerHTML = item.childs[k].text;

if(item.childs[k].marginLeft!=null) span.style.marginLeft = item.childs[k].marginLeft;

if(item.childs[k].marginTop!=null) span.style.marginTop = item.childs[k].marginTop;

if(item.childs[k].className!=null) span.className = item.childs[k].className;

if(item.childs[k]['data-src']!=null) span.setAttribute('data-src',item.childs[k]['data-src']);

ele.appendChild(span);

}else{

showEle(item.childs[k],ele);

}

};

}

pa.appendChild(ele);

}

//Html结构转字符串形式显示 支持<br>换行

function ToHtmlString(htmlStr) {

return toTXT(htmlStr).replace(/\&lt\;br[\&ensp\;|\&emsp\;]\*[\/]?\&gt\;|\r\n|\n/g, "<br/>");

}

//Html结构转字符串形式显示

function toTXT(str) {

var RexStr = /\<|\>|\"|\'|\&|　| /g

str = str.replace(RexStr,

function (MatchStr) {

switch (MatchStr) {

case "<":

return "&lt;";

break;

case ">":

return "&gt;";

break;

case "\"":

return "&quot;";

break;

case "'":

return "&#39;";

break;

case "&":

return "&amp;";

break;

case " ":

return "&ensp;";

break;

case "　":

return "&emsp;";

break;

default:

break;

}

}

)

return str;

}

―――Lake.js前端开发辅助系统源程序到此全部结束―――

―――Lake.js前端开发工具包(V1.0)源程序共82页，到此全部结束―――