**click事件**

举例：点击graphic,然后从graphicsLayer中删除

dojo.connect(map.graphics, "onClick", function(e){

       var graphic=evt.graphic;

       map.graphics.remove(graphic);

});

## dojo/\_base/connect

This module defines the dojo.connect API. This modules also provides keyboard event handling helpers. This module exports an extension event for emulating Firefox's keypress handling. However, this extension event exists primarily for backwards compatibility and is not recommended. WebKit and IE uses an alternate keypress handling (only firing for printable characters, to distinguish from keydown events), and most consider the WebKit/IE behavior more desirable.

这个模块提供了键盘事件操作，也包含一些效仿火狐下压操作的事件。然而，这些扩展事件一般都是向下兼容并且一般都不做推荐。浏览器工具包和IE仅能够对为了识别键盘操作事件而打印输出字符串来进行自动处理。

/\*

function initToolbar() {

alert("into initToolbar");

tb = new Draw(map);

tb.on("draw-end", changeIcon);

on(dom.byId("info"), "click", function(evt) {

if ( evt.target.id === "info" ) {

return;

}

var tool = evt.target.id.toLowerCase();

map.disableMapNavigation();

tb.activate(tool);

});

}

//增加按钮事件

function changeIcon() {

alert("into changeIcon");

//deactivate the toolbar and clear existing graphics

tb.deactivate();

map.enableMapNavigation();

// figure out which symbol to use

var symbol;

//定义鼠标事件

var evt = new MouseEvent("click", {

bubbles: true,

cancelable: true,

view: window

});

var cb = document.getElementById("change\_icon"); //element to click on

/\*

arrayUtils.forEach(points, function(point) {

var graphic = new Graphic(new Point(point), createSymbol(iconPath, initColor));

//console.log(createSymbol(iconPath, initColor));

//var graphic = new Graphic(new Point(point), createSymbol(initColor));

map.graphics.add(graphic);

console.log(point);

});

\*/

//map.graphics.add(new Graphic(evt.geometry, symbol));

}

\*/

涉及到表格参数的读取：  
