SDU Document Format 实验报告

——基于 WEB 的在线编辑器 SDU Online Editor

学院：软件学院

专业：软件工程

课程名：现代软件开发技术

项目：基于 WEB 的在线编辑器（SOE）

组长：熊凤超

组员：张超、燕红磊、王若希、刘琦

2013 年 5 月

目录

一、 任务分工

姓名	学号	任务
熊凤超	201000301233	统筹组织，需求分析，系统设计，代码编写
张超	201000301265	前期负责，需求分析，系统设计，代码编写
燕红磊	201000301244	中期负责，系统设计，代码编写，功能优化
王若希	201000301196	代码编写，文档编写，功能优化，系统测试
刘琦	201000301124	后期负责，代码编写，系统测试，文档编写

二、 系统开发平台

开发工具： JetBrains WebStorm 6.0

开发语言：JavaScript、HTML 和 CSS

操作系统： Windows 7

三、 需求分析

1. 任务陈述

1.1 SDU Document Format 项目整体介绍

基于 Unicode，设计能够基于云端扩展的字符编码和字库 SDU Extensional Unicode（SEU）；基于 XML 格式，设计适用于现代文和古籍文档保存的文档格式 SDU Document Format（SDF）。基于 SEU 和 SDF，完成字库的实现、云输入法的实现；文档编辑、保存；文档上传至服务器，以及在服务器的增删改查。

1.2 模块划分与分工情况：

1. 标准制定
2 基于 XML 的系统开发
3 文档编辑器的开发
   3.1 所见即所得的文档编辑器 SDU Document Editor（SDE）
   3.2 基于 WEB 的在线编辑器 SDU Online Editor（SOE）
4. 云端的开发
5 云端输入法的开发（备选）

本组成员负责3.2阶段的开发工作，即**基于WEB的在线编辑器SDU Online Editor（SOE）**。

2. 任务目标

参考最流行文档编辑软件Microsoft Word的基本功能和整体布局，在浏览器端设计出在线文档编辑器，满足用户文本输入、文本编辑和修饰、表格、项目符号等元素的使用、文本布局的设置等常用功能，使之成为简单易用、操作便捷的在线文档编辑软件。

3 功能设计

3.1 文本输入。

3.2 文本编辑（修饰），能够设置文字的基本属性如：大小，颜色，字体，加粗，下划线，斜体，背景色等。

3.3 元素的使用，能够在编辑器中插入表格，超链接，编号，项

目符号等。

3.4 布局设置，能够设置段落格式包括列表，对齐方式及缩进。

4 系统需求说明

4.1 软件环境

浏览器（IE/Firefox/ Google Chrome/Opera 等）

4.2 硬件环境

Pentium(R) 4 CPU 1.80GHz　　1.82 GHz, 504MB 内存

四、 系统设计

1. 整体设计

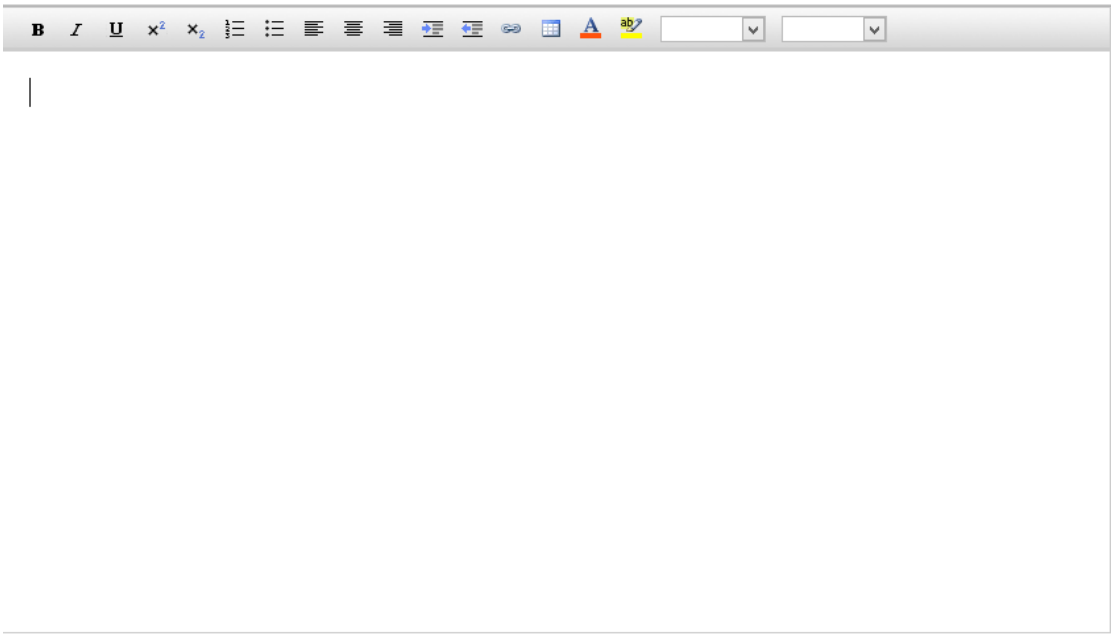
Js	一些 js 文件
Css	一些 css 文件
Icons	一些图标的图片

2. 模块设计

edit_icons.css	功能图标的 CSS
main.css	界面 CSS
base.js	基础的功能（获得元素内容，获得某个元素的前一个元素，获得元素的后一个元素等）js
core.js	样式的修改，编辑框的事件绑定的 js
editor.js	隐藏颜色选择框，插入超链接，插入表格，设置颜色，改变文字大小等功能 js

3. 界面设计

主界面是一个文本编辑器上面有一些设置的按钮，具体如下



4. 主要代码

1. base.js 关键代码实现

1.1 setSelect（）函数

此功能是设置选中内容，首先判断浏览器的类型，首先判断光标的起始位置，然后获得节后的位置最后在结束点重叠，具体代码如下：

```
function setSelect(startTag, endTag, start, end, collapse, type) {
    var Range,selection;

    if(!startTag)
        return false;

    if(document.body.createTextRange){//IE浏览器
        end = end > 0 ? end - 1 : end;
        Range = document.body.createTextRange();
        Range.moveToElementText(startTag);
        Range.moveStart('character', start);
        Range.moveEnd('character', end);
        if(collapse)Range.collapse(false);//在结束点重叠
        Range.select();//显示光标
    }else{//w3c
        range = document.createRange();

        if(browser().GECKO && 3 == type){
            range.setStart(startTag.firstChild, start);
            range.setEnd(endTag.firstChild, endTag.innerHTML.length);
        }else{
            range.setStart(startTag, start);
            range.setEnd(endTag, end);
        }

        if(collapse)range.collapse(false);
        selection = window.getSelection();
        selection.removeAllRanges();
        selection.addRange(range);
    }
}
```

1.2 changeLevel1Style（）修改一级标签的样式

此功能是修改一级标签样式，首先判断当前的标签是否有 textIndent 的 Css 如果有的话那么就直接修改它的值，如果没有就设置其 CSS 的值为传递过来的 ValueItem 具体代码如下：

```
function changeLevel1Style (tag, StyleItem, ValueItem) {
    var textIndentNum = "";
    if(hasStyle(tag, 'textIndent') && StyleItem == 'textIndent'){
         var tempNum = parseInt(tag.style[StyleItem]) + parseInt(ValueItem);
         if(tempNum >= 0)
            tag.style[StyleItem] = tempNum + "em";
         else
            tag.style[StyleItem] = parseInt(tag.style[StyleItem]) + "em";
    }else
        tag.style[StyleItem] = ValueItem;
}
```

1.3 setLine (children, id) 设置 ul,ol 过行

首先创建一个变量 pTag 其中 pTag 中包含 p 标签、span 标签以及当前列表的序列值，然后遍历传递过来要修改的标签的 children，然后以此用 ptag 替换 children 中的节点。替换完毕之后判断是否是在 ul 或者 ol 的结尾跳转如果不是那么对应删除 ptag，具体代码如下：

```
function setLine (children, id) {
    var sign = false;
    var pTag = document.createElement("p");

    for(var i = children.length - 1; i >= 0; i--){
        var tagName = children[i].nodeName.toLowerCase();

        pTag.setAttribute('id', ++id);
        pTag.innerHTML = browser().IE ? "<span></span>" : "<span><br /></span>";;
        if(tagName == "br"){
            children[i].parentNode.replaceChild(pTag, children[i]);
            sign = true;
            break;
        }else if((tagName == "p" && children[i].innerText == "") || tagName == "div"){
            children[i].parentNode.replaceChild(pTag, children[i]);
            sign = true;
            break;
        }
    }

    if(sign){
       if(pTag.nextSibling != null && pTag.previousSibling.nodeName == pTag.nextSibling.nodeName){
           pTag.previousSibling.appendChild(pTag.cloneNode(true));
           pTag.previousSibling.innerHTML += pTag.nextSibling.innerHTML;
           pTag.parentNode.removeChild(pTag.nextSibling);
           pTag.parentNode.removeChild(pTag);
       }
       setSelect($byId(id), $byId(id), 0, 0, true, 1);
       return true;
    }

   return false;

```

1.4 overLay 弹出一个对话框的层

此功能为弹出一个透明层出来通过在 body 标签后添加一个 DIV 标签出来然后设置 div 的长宽透明度和位置等基本属性为添加表格等需要弹出层的功能做准备，具体代码如下：

```
//遮罩层，即弹出一个层出来
function overLay (display, display2, sh, sw) {
    var h = $(document).height();
    var w = $(document).width();

    $(function() {

        $("body").append("<div id='overlay'></div>");

            .height(h)
            .css({
                'opacity' : 0.4,
                'position': 'absolute',
                'top': 0,
                'left': 0,
                'background-color': '#eee',
                'display': display,
                'width': '100%',
                'z-index': 5000
            });
    });

    $(function() {

        $("body").append("<div id='overlay2'></div>");

        $("#overlay2")
            .height(sh + 10)
            .width(sw + 10)
            .css({
                'opacity' : 0.4,
                'position': 'absolute',
                'top': (h - sh)*0.4 - 5,
                'left': (w - sw)*0.5 - 5,
                'background-color': '#ccc',
                'display': display2,
                'z-index': 5001
            });
    });
}
```

2. core.js 关键代码实现

2.1 function ew(id)

设置一个类为 ew 然后设置它的一些基本属性和方法，然后通过原型的方式设置一些方法如是否开启编辑、获取选择的内容、重新设置选中的内容，具体代码如下：

```
function ew(id){
    this.startId = 1;//标签id的起始号
    this.target = $byId(id);//编辑框
    this.selection = null;//选中的内容
    this.selparent = null;//父节点
    this.contain = document.createElement('div');//选中内容副本容器
};
```

2.2 init()初始化 onclick 和 onkeyup 的事件

设置指定目标的 1.1 onclick 和 onkeyup 的事件具体代码如下：

```
//初始化
ew.prototype.init = function (){
    var _self = this;//用于作为事件传递参数
    this.target.onkeyup = function(e){
        try{
            _self.keyboardEventUp(e, _self);
        }catch(ep){
            return false;
        }
    };
    this.target.onclick = function(e){
        try{
            _self.linkClick(e);
            _self.setCmdState(e, _self);
        }catch(ep){
            return false;
        }
    };
    this.editable(true);//开启编辑功能
```

2.3 getSelection()获取选择的内容

```
ew.prototype.getSelection = function (){
    if(browser().IE && 8 == browser().VERSION) this.target.focus();
    if(window.getSelection){//w3c

        this.selection = window.getSelection().getRangeAt(0);
        this.selparent = this.selection.commonAncestorContainer;
        if(this.selparent.nodeType == 3){
            this.selparent = this.selparent.parentNode;
        }
        this.contain.innerHTML = "";
        this.contain.appendChild(this.selection.cloneContents());

    }else if(document.selection && document.selection.createRange){//ie

        this.selection = document.selection.createRange();
        this.selparent = this.selection.parentElement();
        this.contain.innerHTML = this.selection.htmlText;
    }

    if(outHTML(this.target).indexOf(outHTML(this.selparent)) < 0)//选择的不是编辑框的内容
        return false;
    else
        return true;
};
```

2.4 keyboardEventUp（）修改跳出 ol,ul 等标签而出现的一级标签不同步现象

通过判断按键的类型做出相应的处理，如果为回车那么就自动设置国行即设置列表项，如果按键为上下左右那么就设置当前的状态为传递过来的状态。具体代码如下：

```
    var e = window.event || arguments[0];
    var kCode = e.keyCode;

    if(13 == kCode){
        var children = _self.target.childNodes;
        if(!setLine(children, _self.startId)){
            var tdTags = _self.target.getElementsByTagName("td");
            for(var i = 0; i < tdTags.length; i++){
                if(setLine(tdTags[i].childNodes, _self.startId))
                    break;
            }
        }
    }

    if(kCode == 37 || kCode == 38 || kCode == 39 || kCode == 40){
        _self.setCmdState(e, _self);
    }

    _self.setId(_self.target, []);
};
```

2.5 resetSelection（）函数整理选中的内容

首先创建一个新的 div 并且设置其相应的内容主要代码如下：

```
var tempDiv = document.createElement('div');
if(tag.innerHTML == ""){
    if(!isNeed){
        var newSpan = this.ct('span');
        tempDiv.appendChild(newSpan);
        replaceHTML(tempDiv, this.selection);
        this.contain.innerHTML = outHTML($byId(newSpan.id));
        var ptag = $byId(newSpan.id).parentNode;
        if(ptag.nodeName.toLowerCase() == 'span'){
            this.apartTag(ptag);
        }
        this.addLi(tag);
    }
    return;
}
```

然后处理第一个文本节点主要代码如下

```
if(firstTag != null){
        var ctParent = firstTag.parentNode;//在容器中的父节点
        var tempTextCon = document.createElement('span');
        tempTextCon.appendChild(firstTag.cloneNode(true));
        if(firstTag.parentNode != tag){
            var realParent = document.getElementById(ctParent.id);//真正的父节点
            var realName = realParent.nodeName.toLowerCase();
            if(realParent.innerHTML != tempTextCon.innerHTML){//内容一样的时候可进行统一处
                var newSpan = realName == 'span' ? realParent.cloneNode(true) : document.c
                newSpan.id = ++this.startId;
                var len = ctParent.innerHTML.length;
                newSpan.innerHTML = firstTag.nodeValue;
                tempDiv.appendChild(newSpan.cloneNode(true));
                var headHtml = realParent.innerHTML.substr(0, realParent.innerHTML.length
                if(realName == 'span'){
                    realParent.innerHTML = headHtml;
                    insertAfter(newSpan.cloneNode(true), realParent);
                    ctParent.parentNode.replaceChild(newSpan, ctParent);
                }else{
                    realParent.innerHTML = headHtml +
                        ctParent.innerHTML.replace(firstTag.nodeValue, tempDiv.innerHTML);
                    firstTag.parentNode.replaceChild(newSpan, firstTag);
                }
                firstTag = newSpan.firstChild;
            }
        }else{
            var nspan = this.selparent.nodeName.toLowerCase() == 'span' ? this.selparent.c
            nspan.id = ++this.startId;
            nspan.innerHTML = "";
            nspan.innerHTML = firstTag.nodeValue;


            if(!nextTag(firstTag)){//只有一个文本标签
                tempDiv.appendChild(nspan.cloneNode(true));
                replaceHTML(tempDiv, this.selection);
            }else{
                var realfirst = $byId(nextTag(firstTag).id).previousSibling;
                realfirst.nodeValue = strReverse(strReverse(realfirst.nodeValue).replace(s
                insertAfter(nspan.cloneNode(true), realfirst);
                //realfirst.parentNode.replaceChild(nspan.cloneNode(true), realfirst);
            }
            tag.replaceChild(nspan, firstTag);
            firstTag = nspan.firstChild;
            if($byId(nspan.id).parentNode.nodeName.toLowerCase() == 'span'){//需要进行span
                this.apartTag($byId(nspan.id).parentNode);
            }
        }
    }

    var lastTag = tag.lastChild;

    while(lastTag && lastTag.nodeType != 3){
        lastTag = lastTag.lastChild;
    }

```

接着处理第二个节点

```
if(firstTag != lastTag && lastTag != null){
        var ctParent = lastTag.parentNode;//在容器中的父节点

        if(lastTag.parentNode != tag){
            var realparent = document.getElementById(ctParent.id);//真正的父节点
            var realName = realparent.nodeName.toLowerCase();
            var tempTextCon = document.createElement('span');
            tempTextCon.appendChild(lastTag.cloneNode(true));
            if(realparent.innerHTML != tempTextCon.innerHTML){//内容一样的时候可进行统一处
                var newSpan = realName == 'span' ? realparent.cloneNode(true) : document.c
                newSpan.id = ++this.startId;
                newSpan.innerHTML = lastTag.nodeValue;
                tempDiv.innerHTML = "";
                tempDiv.appendChild(newSpan.cloneNode(true));
                var len = ctParent.innerHTML.replace(/[\t\n\x0B\f\r]/gi, "").length;
                if(realName == 'span'){
                    realparent.innerHTML = realparent.innerHTML.replace(lastTag.nodeValue,
                    realparent.parentNode.insertBefore(newSpan.cloneNode(true), realparent
                    ctParent.parentNode.replaceChild(newSpan, ctParent);
                }else{
                    realparent.innerHTML = realparent.innerHTML.replace(/[\t\n\x0B\f\r]/gi
                    lastTag.parentNode.replaceChild(newSpan, lastTag);
                }
            }
        }else{//父标签为容器
            var nspan = this.ct('span');
            nspan.innerHTML = lastTag.nodeValue;

            if(!previousTag(lastTag)){//只有一个文本标签
                tempDiv.appendChild(nspan);
                replaceHTML(tempDiv, this.selection);
            }else{
                var reallast = $byId(previousTag(lastTag).id).nextSibling;
                reallast.nodeValue = reallast.nodeValue.replace(lastTag.nodeValue, "");
                insertAfter(nspan.cloneNode(true), $byId(previousTag(lastTag).id));
            }

            tag.replaceChild(nspan, lastTag);
            if($byId(nspan.id).parentNode.nodeName.toLowerCase() == 'span'){//需要进行span
                this.apartTag($byId(nspan.id));
            }
        }
    }
    this.addSpan(tag);
    this.addLi(tag);
```

2.6 setCmdState 设置按钮的选中状态

首先获得 span 标签 li 标签和 p 标签

```
var spanArr = _self.contain.getElementsByTagName('span');//span标签
var liArr = _self.contain.getElementsByTagName('li');//li标签
var pArr = _self.contain.getElementsByTagName('p');//p标签
```

然后依次对三种标签做相应的处理

```
if(spanArr.length == 0){
    var selpname = _self.selparent.nodeName.toLowerCase();
    liArr = [];
    pArr = [];
    spanArr = [];
    if(selpname == 'span'){
        spanArr.push(_self.selparent.cloneNode(true));
        var selppname = _self.selparent.parentNode.nodeName.toLowerCase();
        if(selppname == 'li' || selppname == 'p')
            selppname == "li" ? liArr.push(_self.selparent.parentNode.cloneNode(true))

    }else if(selpname == 'li' || selpname == 'p'){
        selpname == "li" ? liArr.push(_self.selparent.cloneNode(true)) : pArr.push(_se
    }
}
```

```
if(pArr.length == 0 && isSameParent(liArr)){
    cmdBtnStyle($byId(liArr[0].id).parentNode.nodeName.toLowerCase() == 'ol' ? 'ol' :
}
```

接着判断设置字体和字样的按钮是否为真正的选中状态，

```
var generalStyleArr = cmdStyleItem.slice(0, 8).concat('fontSize', 'fontFamily');

for(var i = 0; i < generalBtnArr.length; i++){
    var cmdId = generalBtnArr[i];//控件的id
    var cmdStyle = generalStyleArr[i];//当前要检测的样式
    var sign = false, sign2 = false;//标志位,sign2为一级标签

    if(cmdStyle == 'fontSize' || cmdStyle == 'fontFamily'){
        for(var j = 0; j < spanArr.length; j++){
            sign = false;
            if(spanArr[j].style[cmdStyle] == "" || spanArr[j].style[cmdStyle] != spanArr[0].style[cmdStyle]){
                break;
            }
            sign = true;
        }
    }else{
        for(var j = 0; j < spanArr.length; j++){
            sign = false;
            if(spanArr[j].style[cmdStyle] == "" || spanArr[j].style[cmdStyle] != cmdValueItem[i][0]){
                break;
            }
            sign = true;
        }
    }
```

```
for(var j = 0; j < liArr.length; j++){
    sign2 = false;

    if(liArr[j].style[cmdStyle] == "" || liArr[j].style[cmdStyle] != cmdValueItem[i][0]){
        break;
    }
    sign2 = true;
}

for(var j = 0; j < pArr.length; j++){
    sign2 = false;

    if(pArr[j].style[cmdStyle] == "" || pArr[j].style[cmdStyle] != cmdValueItem[i][0]){
        break;
    }
    sign2 = true;
}
```

最后如果为选中状态那么就设置相应的字体和字样

```
    if(sign || sign2){

        if(cmdStyle == 'fontSize'){//字体大小状态设置
            var opts = $byId('fS').getElementsByTagName('option');
            for(var k = 0; k < opts.length; k++){

                if(opts[k].value == parseInt(spanArr[0].style['fontSize']) || opts[k].value == spanArr[0].style
                    opts[k].selected = true;
                }
            }
        }else if(cmdStyle == 'fontFamily'){//字体状态设置
            var opts = $byId('fF').getElementsByTagName('option');
            for(var k = 0; k < opts.length; k++){

                if(opts[k].value.toLowerCase() == spanArr[0].style['fontFamily'].toLowerCase()){
                    opts[k].selected = true;
                }
            }
        }else
            cmdBtnStyle(cmdId, '#8397b2', '#d5e2f3');
    }
}
```

2.7 paraChange（）对段落的修改

首先调用 resetSelection 重新整理选中的内容，然后依次对 li p span 标签做处理。具体代码如下：

```
lew.prototype.paraChange = function (style, value){
    if(!this.getSelection())return false;
    this.resetSelection(this.contain);
    var liArr = this.contain.getElementsByTagName('li');
    var pArr = this.contain.getElementsByTagName('p');
    var spanArr = this.contain.getElementsByTagName('span');

    if(liArr.length == 0 && pArr.length == 0){
        liArr = [];
        liArr.push($byId(spanArr[0].id).parentNode.cloneNode(true));
    }

    for(var i = 0; i < liArr.length; i++){
        changeLevel1Style($byId(liArr[i].id), style, value[0]);
    }

    for(var j = 0; j < pArr.length; j++){
        changeLevel1Style($byId(pArr[j].id), style, value[0]);
    }
    this.reSelect();
};
```

3 editor.js 中关键代码

3.1 基本变量的设置

设置按钮的类型，按钮所对应的 css 属性、按钮的值以及按钮的的处理类的变量等。具体代码如下：

```
var curEw;
var cmdIdItem = ['b', "i", "u", "superscript", "suffix", "algLf", "algCn",
                 "algRg", "addIndtat", "reduceIndtat", "ol", "ul"];//控制项id数组
var cmdTypeItem = [1, 1, 1, 1, 1, 3, 3, 3, 3, 3, 2, 2];//控制项类型['8px', '8px'], ['Arial', 'Arial'],
var cmdValueItem = [['700', 'normal'], ['italic', 'normal'],
                    ['underline', 'none'], ['super', 'baseline'], ['sub', 'baseline'],
                    ['left', 'left'], ['center', 'center'], ['right', 'right'],
                    ['2em', '2em'], ['-2em', '-2em'], ['inherit', 'none'], ['disc', 'none']];//控制项值'fontSize',
var cmdStyleItem = ['fontWeight', 'fontStyle', 'textDecoration',
                    'verticalAlign', 'verticalAlign', 'textAlign', 'textAlign', 'textAlign',
                    'textIndent', 'textIndent', 'listStyleType', 'listStyleType'];

window.onload = function (){
```

3.2 Exct（）函数，执行用户请求

通过判断按钮的控制类型调用处理类的相应函数做出处理具体代码如下：

```
function exct(type, style, value, tagName){

    switch(type){
        case 1:
            return function(){curEw.fontStyleChange(style, value); curEw.setCmdState(null, curEw);}

        case 2:
            return function(){curEw.ouListChange(style, value, tagName); curEw.setCmdState(null, curEw);}

        case 3:
            return function(){curEw.paraChange(style, value); curEw.setCmdState(null, curEw);}
    }
}
```

3.3 sureLink()函数插入链接确认

首先判断插入的链接是否合法

```
if(linkText.value == ""){
    $byId('tip').innerHTML = '显示的文本不能为空！';
    sign = false;
}else if(!isUrl(linkAddress.value)){
    $byId('tip').innerHTML = '您输入的链接地址有误！';
    sign = false;
```

然后设置插入链接的 css 并且把 url 插入到当前页面，最后使弹出来的层消失

```
}else if(linkText.getAttribute('disabled') != null && linkText.getAttribute('disabled').toLowerCase() == 'disabl
    for(var i = 0; i < spanArr.length; i++){
        $byId(spanArr[i].id).style['color'] = 'blue';
        $byId(spanArr[i].id).style['textDecoration'] = 'underline';
        $byId(spanArr[i].id).setAttribute('title', titleText);
        $byId(spanArr[i].id).setAttribute('name', 'link');
    }
}else{
    insertAfter(tempA, $byId(spanArr[0].id));
    for(var i = spanArr.length - 1; i >= 0; i--){
        $byId(spanArr[i].id).parentNode.removeChild($byId(spanArr[i].id));
    }
}
if(sign){
    linkDialog('none', 'none');
}
```

3.4 sureTable ()插入表格确认

首先获得弹出对话框输入的行数和列数

```
var rs = $byId('rs').value;
var cs = $byId('cs').value;
var tb = curEw.ct('table');
var tbody = curEw.ct('tbody');
var np = null;
var spanArr = curEw.contain.getElementsByTagName('span');
```

然后根据输入的行数和列数把表格插入到当前的页面并且设置好相应的 Css

```
    for(var i = 0; i < rs; i++){
        var tr = curEw.ct('tr');

        for(var j = 0; j < cs; j++){
            var td = curEw.ct('td');
            var newp = curEw.ct('p');
            var newspan = curEw.ct('span');
            newspan.innerHTML = browser().IE ? "" : "<br />";
            newp.appendChild(newspan);
            td.appendChild(newp);
            tr.appendChild(td);
        }

        tbody.appendChild(tr);
    }
    tb.appendChild(tbody);
    $byId(spanArr[0].id).appendChild(tb);
    var tbp = $byId(spanArr[0].id);
    var tbpp = $byId(spanArr[0].id).parentNode;
    curEw.target.innerHTML = curEw.target.innerHTML.replace(outHTML($byId(tb.id)), "</span></"+ tbpp.nodeName +">"
       outHTML($byId(tb.id)) + "<"+ tbpp.nodeName +" id=" + ++curEw.startId + " style="+ tbpp.getAttribute('style')
       ++curEw.startId + " style=" + tbp.getAttribute('style') + ">");
    setSelect($byId(tb.id).getElementsByTagName('span')[0], $byId(tb.id).getElementsByTagName('span')[0], 0, 1, tru
    tableDialog('none', 'none');
}
```

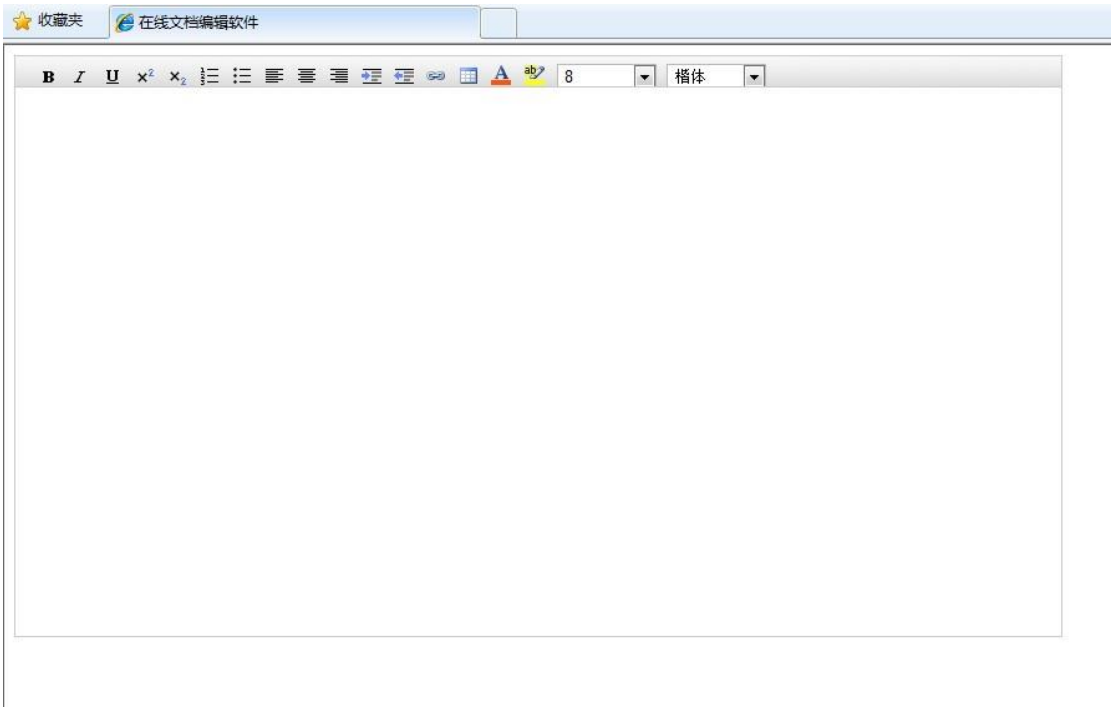
3.5 setColor ()设置字体的颜色和字体的背景颜色

如果是设置字体颜色直接设置其 css 的 color 值即可，如果是设置字体的背景颜色那么直接生成一个块，并且这个块的长宽和选中的字的长宽相同，然后设置这个块的颜色即可。具体代码如下：

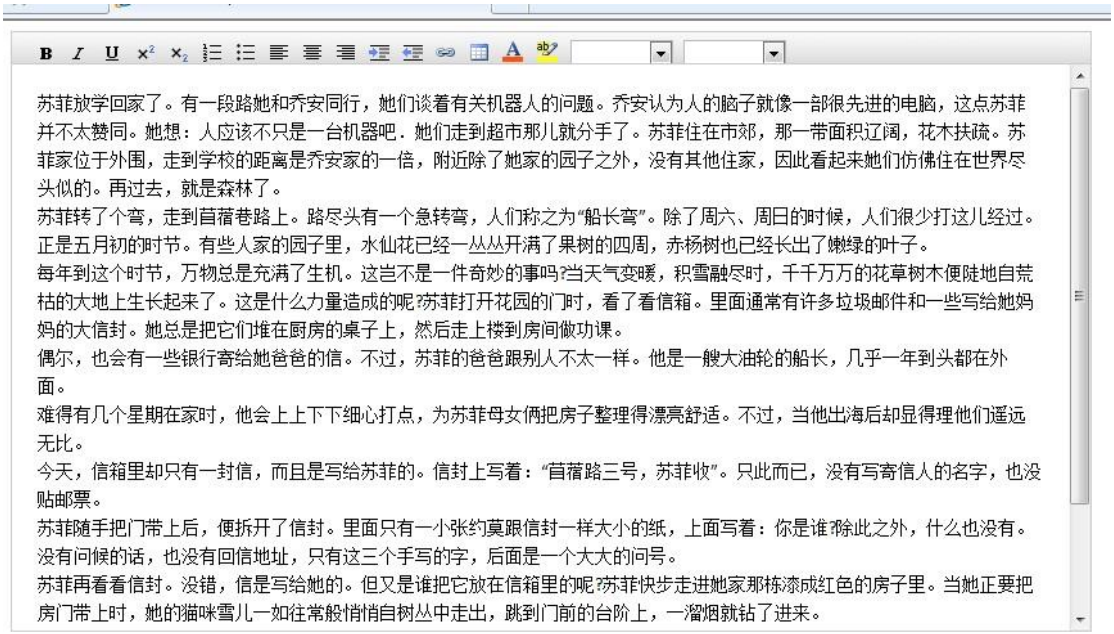
```
function setColor () {

    setCBack();

    var id = this.id;
    var top = $("#" + id).offset().top;
    var left = $("#" + id).offset().left;
    var height = $("#" + id).height();
    var width = $("#" + id).width();

    $('#' + id).css({
        'border-width': '1px',
        'border-color': '#8397b2',
        'border-style': 'solid',
        'background-color': '#d5e2f3'
    });

    $(id == 'fontColor' ? '#fcSel' : '#bcSel').css({
        'top': top + height + 4,
        'left': left,
        'display': 'block',
        'z-index':10000
    });
}

//改变文字大小的value
```

五、 使用说明/测试用例

1. 打开浏览器，输入地址，进入网页。



2. 在文本编辑框中输入文字或粘贴外部文字。



苏菲放学回家了。有一段路她和乔安同行，她们谈着有关机器人的问题。乔安认为人的脑子就像一部很先进的电脑，这点苏菲并不太赞同。她想：人应该不只是一台机器吧．她们走到超市那儿就分手了。苏菲住在市郊，那一带面积辽阔，花木扶疏。苏菲家位于外围，走到学校的距离是乔安家的一倍，附近除了她家的园子之外，没有其他住家，因此看起来她们仿佛住在世界尽头似的。再过去，就是森林了。

苏菲转了个弯，走到苜蓿巷路上。路尽头有一个急转弯，人们称之为"船长弯"。除了周六、周日的时候，人们很少打这儿经过。正是五月初的时节。有些人家的园子里，水仙花已经一丛丛开满了果树的四周，赤杨树也已经长出了嫩绿的叶子。

每年到这个时节，万物总是充满了生机。这岂不是一件奇妙的事吗?当天气变暖，积雪融尽时，千千万万的花草树木便陡地自荒枯的大地上生长起来了。这是什么力量造成的呢?苏菲打开花园的门时，看了看信箱。里面通常有许多垃圾邮件和一些写给她妈妈的大信封。她总是把它们堆在厨房的桌子上，然后走上楼到房间做功课。

偶尔，也会有一些银行寄给她爸爸的信。不过，苏菲的爸爸跟别人不太一样。他是一艘大油轮的船长，几乎一年到头都在外面。

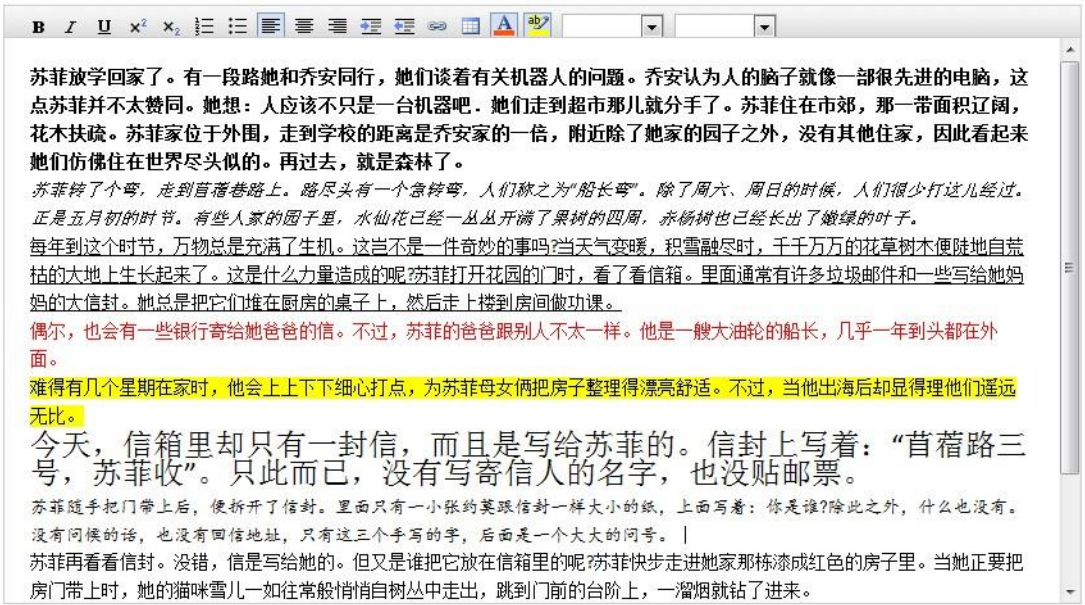
难得有几个星期在家时，他会上上下下细心打点，为苏菲母女俩把房子整理得漂亮舒适。不过，当他出海后却显得理他们遥远无比。

今天，信箱里却只有一封信，而且是写给苏菲的。信封上写着："苜蓿路三号，苏菲收"。只此而已，没有写寄信人的名字，也没贴邮票。

苏菲随手把门带上后，便拆开了信封。里面只有一小张约莫跟信封一样大小的纸，上面写着：你是谁?除此之外，什么也没有。没有问候的话，也没有回信地址，只有这三个手写的字，后面是一个大大的问号。

苏菲再看看信封。没错，信是写给她的。但又是谁把它放在信箱里的呢?苏菲快步走进她家那栋漆成红色的房子里。当她正要把房门带上时，她的猫咪雪儿一如往常般悄悄自树丛中走出，跳到门前的台阶上，一溜烟就钻了进来。

3. 编辑和修饰文字（功能栏从左往右依次为：加粗、斜体、下划线、上标、下标、编号、项目符号、左对齐、居中、右对齐、增加缩进、减少缩进、添加超链接、插入表格、文字颜色、背景颜色、文字大小、字体）



苏菲放学回家了。有一段路她和乔安同行，她们谈着有关机器人的问题。乔安认为人的脑子就像一部很先进的电脑，这点苏菲并不太赞同。她想：人应该不只是一台机器吧．她们走到超市那儿就分手了。苏菲住在市郊，那一带面积辽阔，花木扶疏。苏菲家位于外围，走到学校的距离是乔安家的一倍，附近除了她家的园子之外，没有其他住家，因此看起来她们仿佛住在世界尽头似的。再过去，就是森林了。

苏菲转了个弯，走到苜蓿巷路上。路尽头有一个急转弯，人们称之为"船长弯"。除了周六、周日的时候，人们很少打这儿经过。正是五月初的时节。有些人家的园子里，水仙花已经一丛丛开满了果树的四周，赤杨树也已经长出了嫩绿的叶子。

每年到这个时节，万物总是充满了生机。这岂不是一件奇妙的事吗?当天气变暖，积雪融尽时，千千万万的花草树木便陡地自荒枯的大地上生长起来了。这是什么力量造成的呢?苏菲打开花园的门时，看了看信箱。里面通常有许多垃圾邮件和一些写给她妈妈的大信封。她总是把它们堆在厨房的桌子上，然后走上楼到房间做功课。

偶尔，也会有一些银行寄给她爸爸的信。不过，苏菲的爸爸跟别人不太一样。他是一艘大油轮的船长，几乎一年到头都在外面。

难得有几个星期在家时，他会上上下下细心打点，为苏菲母女俩把房子整理得漂亮舒适。不过，当他出海后却显得理他们遥远无比。

今天，信箱里却只有一封信，而且是写给苏菲的。信封上写着："苜蓿路三号，苏菲收"。只此而已，没有写寄信人的名字，也没贴邮票。

苏菲随手把门带上后，便拆开了信封。里面只有一小张约莫跟信封一样大小的纸，上面写着：你是谁?除此之外，什么也没有。没有问候的话，也没有回信地址，只有这三个手写的字，后面是一个大大的问号。

苏菲再看看信封。没错，信是写给她的。但又是谁把它放在信箱里的呢?苏菲快步走进她家那栋漆成红色的房子里。当她正要把房门带上时，她的猫咪雪儿一如往常般悄悄自树丛中走出，跳到门前的台阶上，一溜烟就钻了进来。

第一段设置文字加粗；

第二段设置文字为斜体；

第三段设置为文字加下划线；

第四段设置文字为**红色**；

第五段设置文字**背景为黄色**；

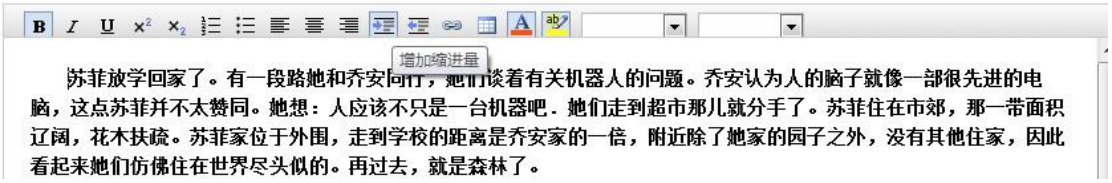
第六段设置文字**大小为 24 号**；

第七段设置字体为“**楷体**”。

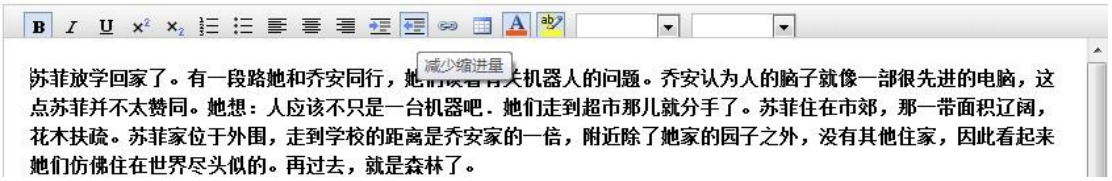
4. 设置特殊格式、段落格式，插入元素

4.1 设置特殊格式（上标和下标）

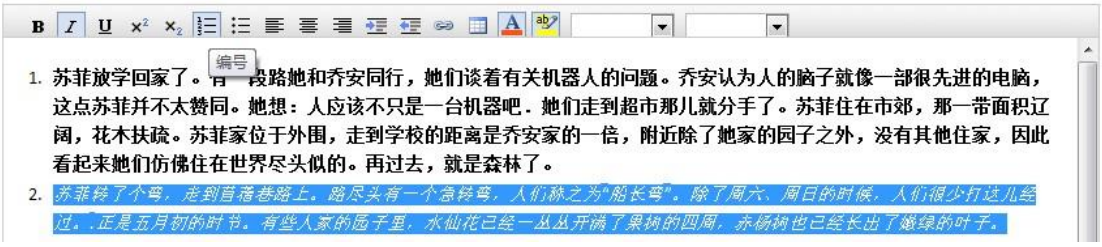
4.2 设置段落格式



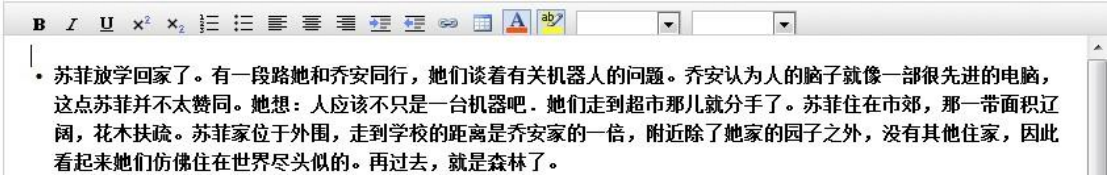
增加缩进



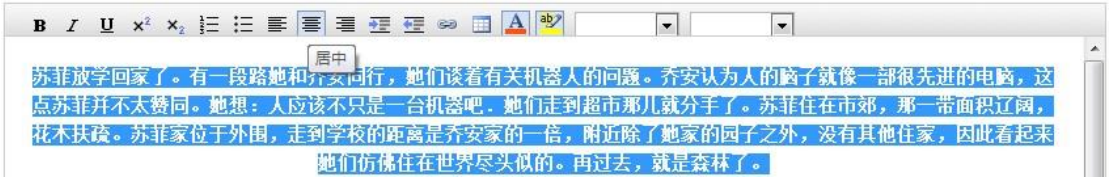
减少缩进



添加编号

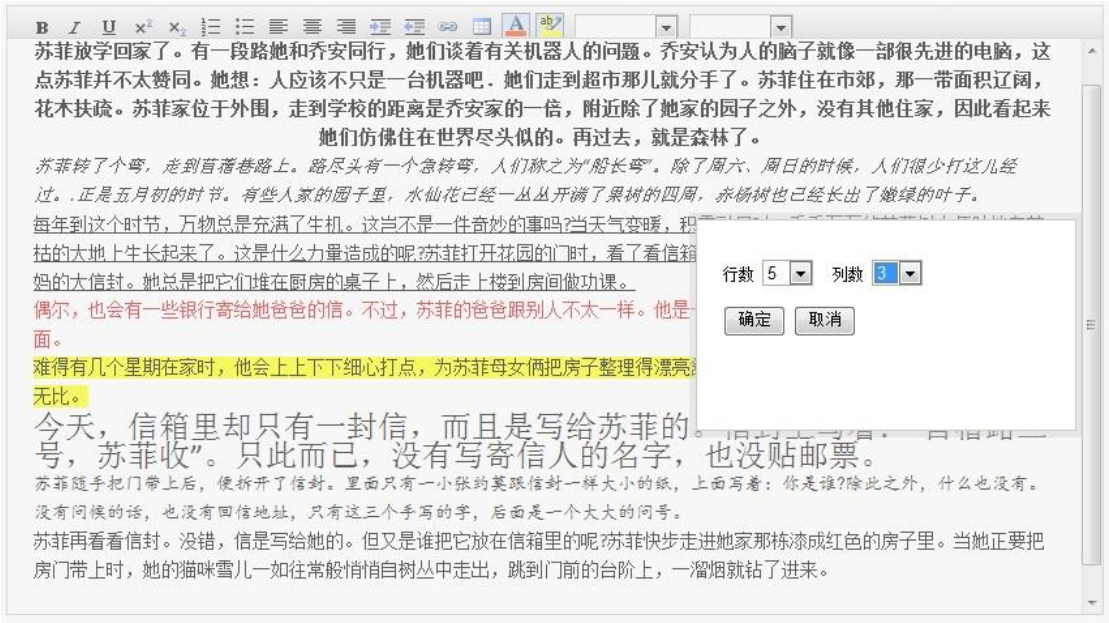


添加项目符号

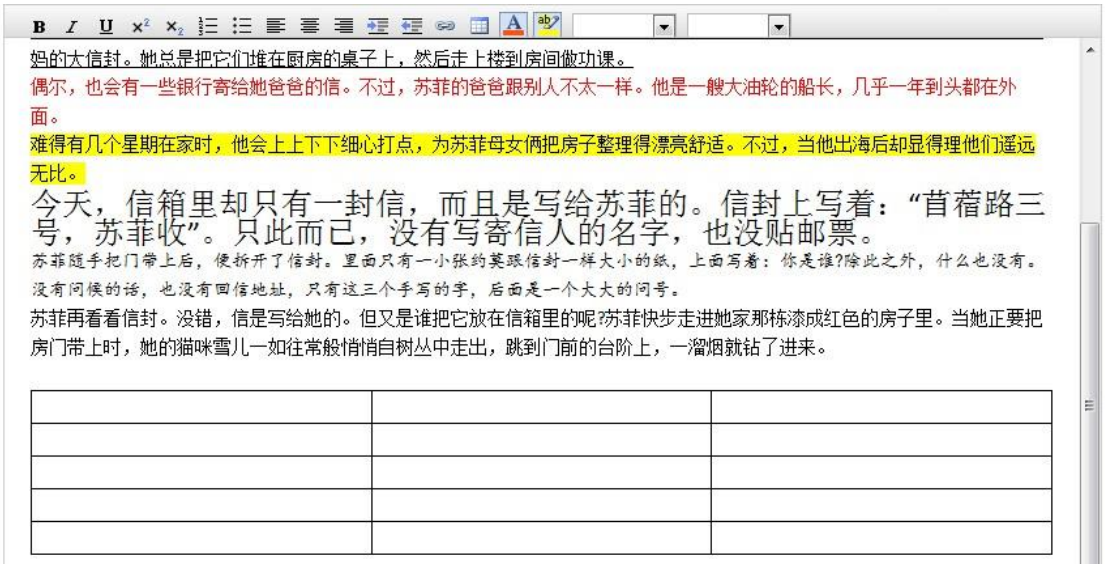


居中

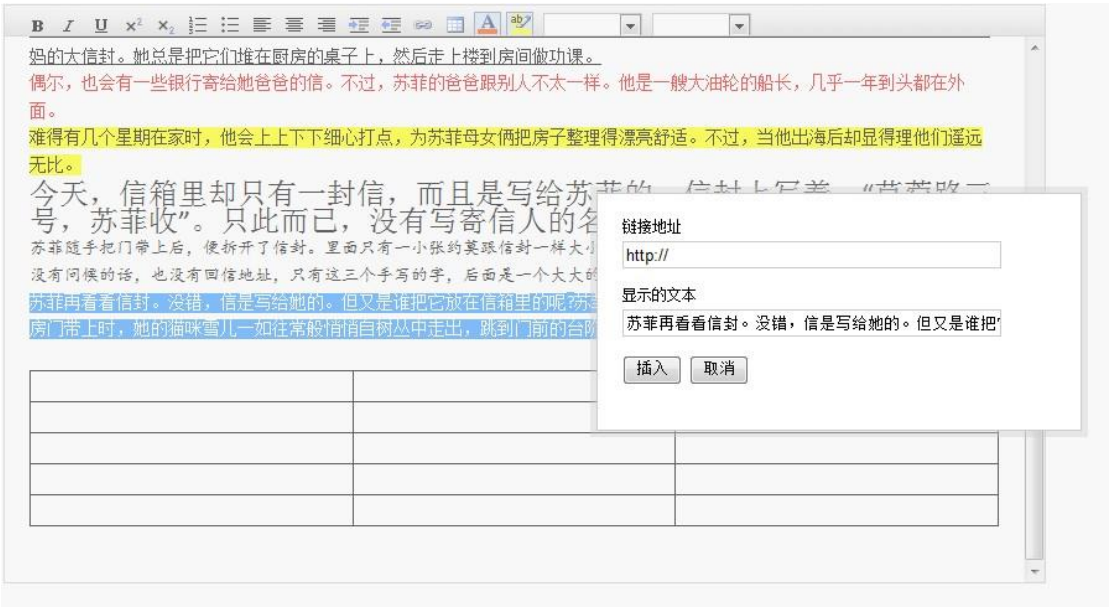
4.3 插入元素



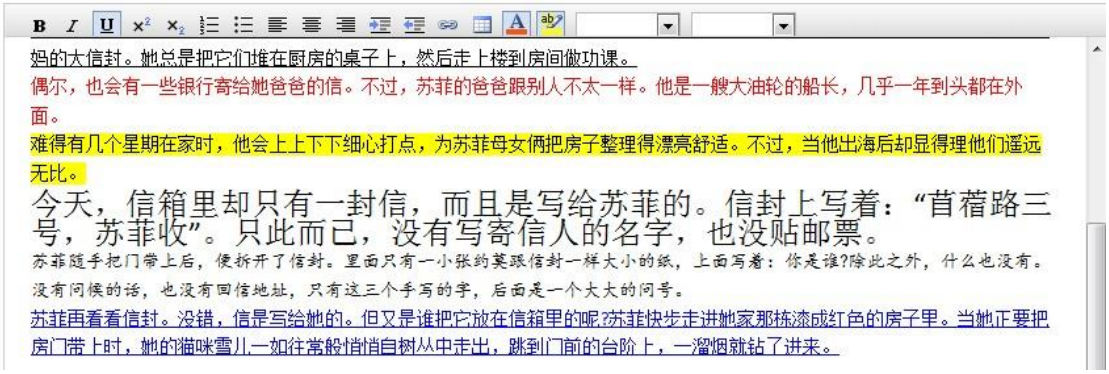
添加表格选项窗口



添加表格成功



添加超级链选项窗口



添加超级链成功

六、 实验总结

基于 WEB 的在线编辑器 SDU Online Editor，作为 SDU Document Format 项目的组成部分之一，既有整体性又有相对独立性。它具备

普通文档编辑器的基本特征,而且要比较轻便和简洁。因此从以上两点出发，本小组展开了这次实验。通过这次实验过程，加深了使用 JavaScript 进行开发的技巧和流程，懂得了团队合作的重要性，对软件开发的规范过程有了更全面的理解和感受，为以后的项目开发打下了基础。