

力软快速开发框架开发示例

前言

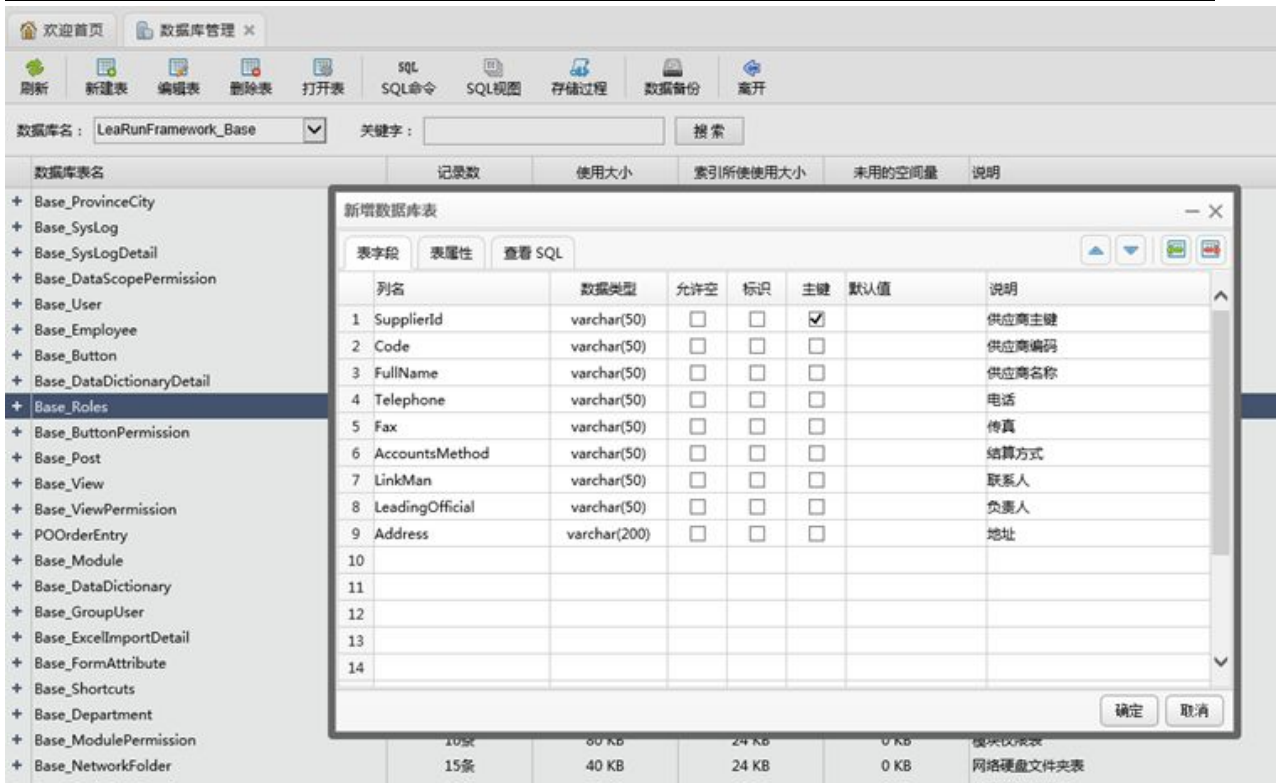
为了使用户快速掌握这套开发框架，作者以开发基础资料中的《供应商》为例，供读者作为开发参考

目录

1、创建数据表.....	1
2、生成代码.....	1
2.1、生成表单面代码.....	1
2.2、设置详细资料页.....	2
2.3、生成页面列表.....	3
2.4、生成设置.....	3
2.5、查看代码.....	4
3、将代码放置到对应的文件夹下.....	4
4、配置系统模块.....	6
5、配置系统按钮.....	7
6、功能授权.....	8
7、代码微调.....	9
7.1、绑定动态数据源的下拉框.....	9
7.2、LR-FDMS 系统常用 JS 库的使用.....	11
7.3、重点代码解读.....	11
7.3.1、表单提交代码.....	11
7.3.2、记录日志.....	11
7.3.3、动态表单.....	11
7.3.4、一个表单中多 tab 处理.....	12

1、创建数据表

打开“系统管理”-“数据库管理”-“新建表”，就可以打开新增表的界面



输入字段、表信息后点确定就完成了新增表。

2、生成代码

打开“系统管理”-“智能开发”-“代码生成”，就可以打开代码生成器界面

这里需要注意下，代码生成器可以支持从 PowerDesign 和数据库两种方式提取表信息，前面我们是从直接建表所以这里需要更改配置谁的里代码生成器连接数据源模式为：database



2.1、生成表单面代码

选择左侧资源管理器中的供应商表，在右侧进行设置。

重点栏位:

合并列：如果哪一个控件比较长就可以在此处设置跨列。

控件验证：表单提交时要进行哪种正则验证。

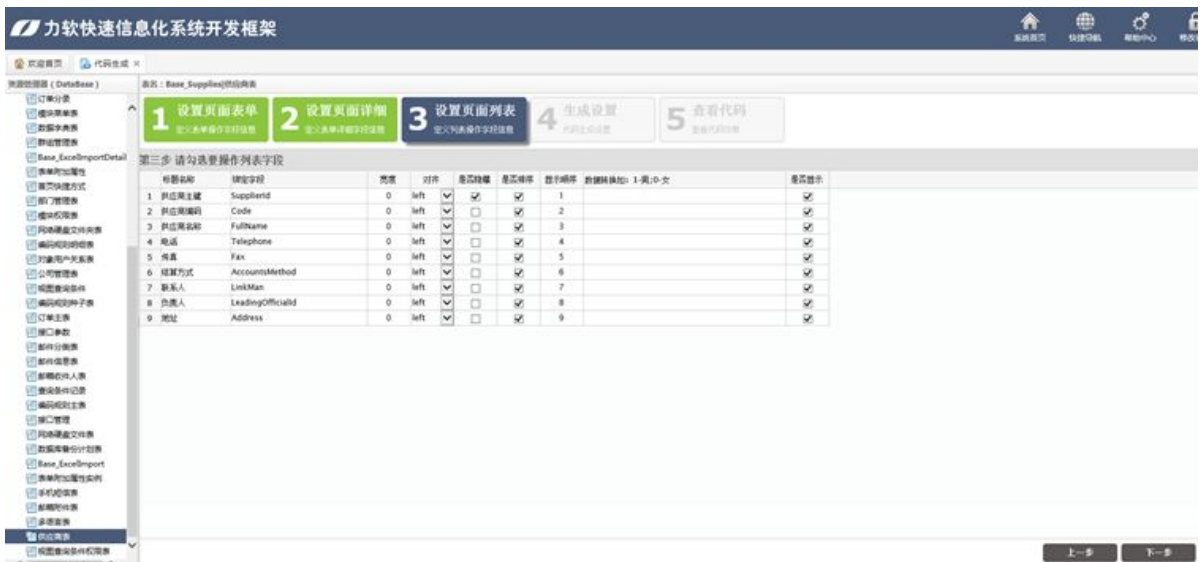
点击下一步进入设置详细资料页

详细资料是不提供编辑的一般是提供给只有查看权限的用户使用，其页面布局跟表单页类似



2.3、生成页面列表

点击下一步进入设置页面列表功能



重要栏位：

是否隐藏：该列是否隐藏起来，一般主键列会隐藏起来。但是在编辑删除操作的时候仍然能取得到主键值，实际上就是把那一列 hidden 了。

数据转换：比如性别字段我们不会存“男”或者“女”而是 0，1 在这里设置一下列表页就能自动转换过来。

2.4、生成设置

点击下一步进行生成设置



重点栏位：

业务区域：该模块的代码文件放置在 MVC 框架中哪一级文件夹下。

排序类型：列表页上的排序类型，asc 升序、desc 降序

排序字段：按哪个字段进行排序。

是否分页：列表页是否分页。

分页大小：每页显示多少条记录。

表单页面：表单页视图的名字。

详细页面：详细页视图的名字。

列表页面：列表页视图的名字。

控制器：控制器的名称。

页面布局：该模块列表页如何布局，这里已经提供了几种方式，用户可以不用去排版。

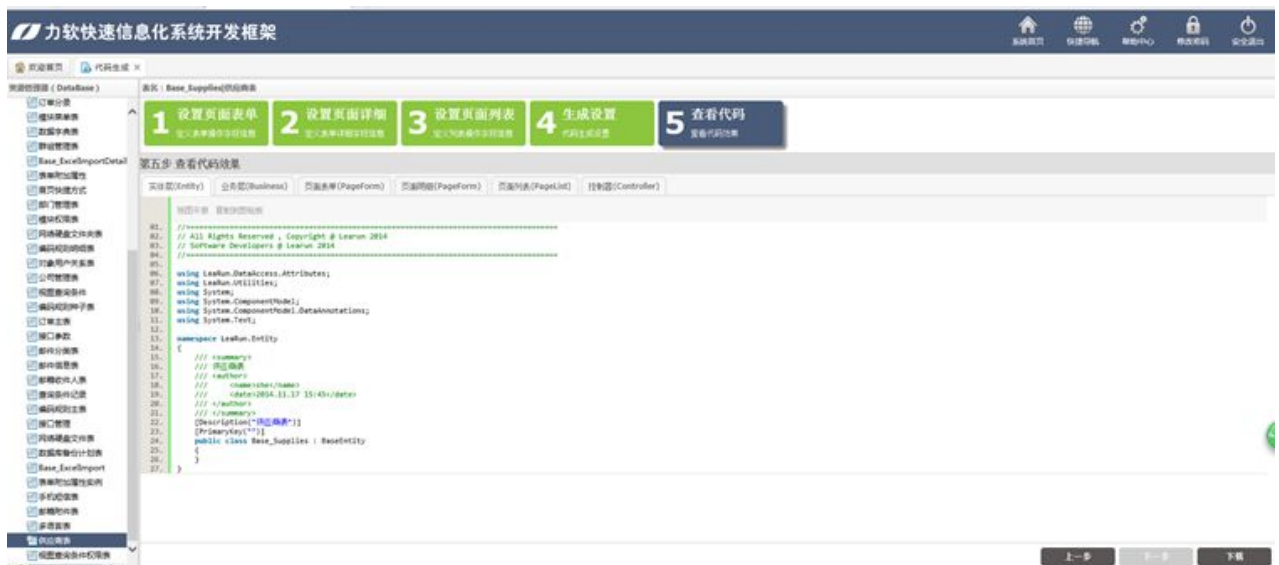
操作：生成增删改查前端脚本，用户可自行修改。

显示列数：表单页里每行显示几列。

2.5、查看代码

点击下一步进入查看代码功能。

到了这一步代码已经生成完毕，在这里可以直接复制或者下载代码。

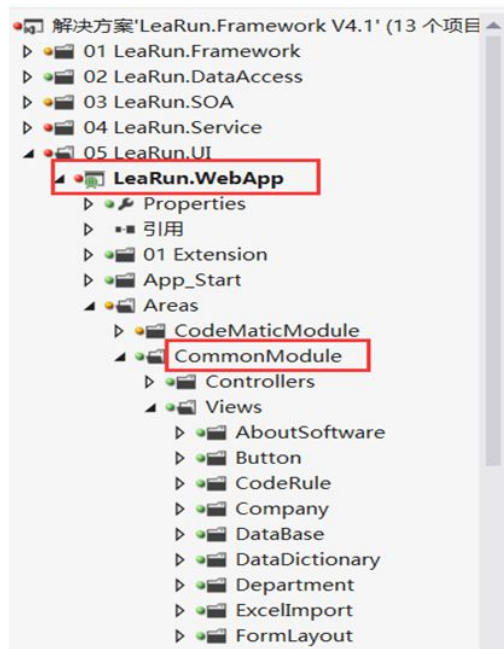


3、将代码放置到对应的文件夹下

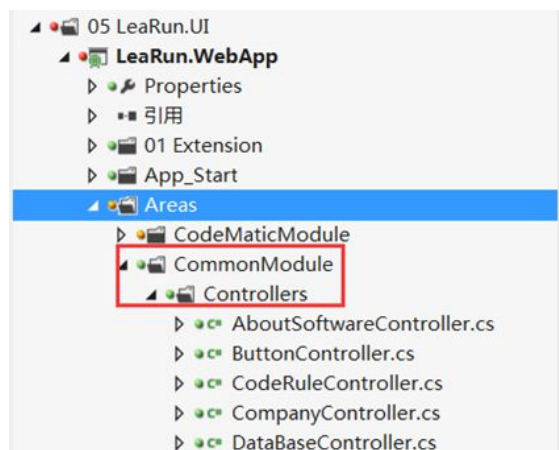
在代码查看页面点击下载按钮把代码下载解压得到如下文件

名称	修改日期	类型	大小
Base_Supplies	2014/11/18 15:13	文件夹	
Base_Supplies.cs	2014/11/18 15:10	CS 文件	3 KB
Base_SuppliesBll.cs	2014/11/18 15:10	CS 文件	1 KB
Base_SuppliesController.cs	2014/11/18 15:10	CS 文件	1 KB

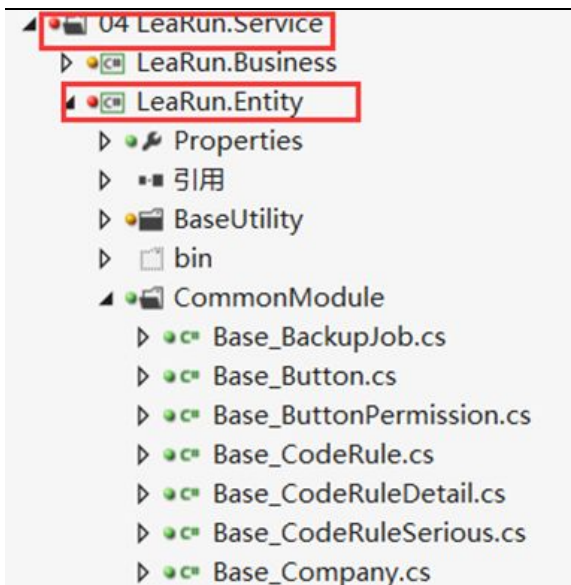
这里 Base_Supplies 文件夹存放的是视图放置在 Areas\ CommonModule\ Views 下



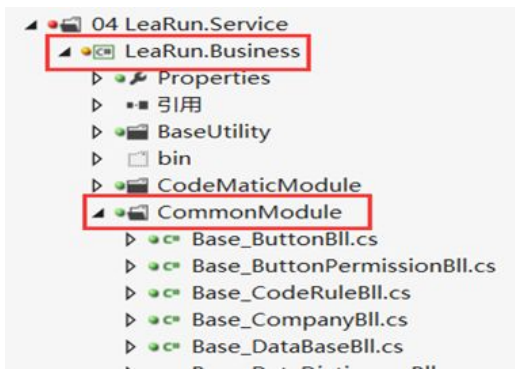
Base_SuppliesController.cs 放置到 Areas\ CommonModule\ Controllers 下



Base_Supplies.cs 放置到 entity 项目下



Base_SuppliesBll.cs 放置在 Business 项目下



4、配置系统模块

代码贴进去以后，要在系统中配置菜单模块，这样刚才做的功能才可以通过菜单访问和授权。
打开“系统管理”-“智能开发”-“系统模块”就进入了模块设置

 力软快速信息化系统开发框架

[欢迎首页](#)

[数据字典](#)

[系统模块](#)

导航目录
 快速开发平台
 系统管理
 基础资料
 公司管理
 部门管理
 角色管理
 岗位管理
 用户组管理
 用户管理
 供应商管理
 业务报表
 案例模块

模块信息

[刷新](#)
[新增](#)
[编辑](#)
[删除](#)
[详细](#)
[离开](#)

	编码	名称	分类	访问地址	目标	层次	有效
1	10.02.01	公司管理	页面	/CommonModule/Company/Index	iframe	2	<input checked="" type="checkbox"/>
2	10.02.02	部门管理	页面	/CommonModule/Department/Index	iframe	2	<input checked="" type="checkbox"/>
3	10.02.03	角色管理	页面	/CommonModule/Roles/Index	iframe	2	<input checked="" type="checkbox"/>
4	10.02.04	岗位管理	页面	/CommonModule/Post/Index	iframe	2	<input checked="" type="checkbox"/>
5	10.02.05	用户组管理	页面	/CommonModule/GroupUser/Index	iframe	2	<input checked="" type="checkbox"/>
6	10.02.06	用户管理	页面	/CommonModule/User/Index	iframe	2	<input checked="" type="checkbox"/>
7	10.02.07	供应商管理	页面	/CommonModule/Supplies/Index	iframe	2	<input checked="" type="checkbox"/>

由于供应商管理是一项基础资料，所以我们选中左侧的基础资料，在他的下级新增子菜单，选中后点击工具栏上的新增按钮

编辑模块信息

模块编码：10.02.07 *

上级模块：基础资料

Icon图标：report_user.png

级别层次：2 *

模块名称：供应商管理 *

模块分类：页面

连接目标：iframe

显示顺序：10027

选项：☒访问权限 ☒数据范围 ☒动态视图 ☒动态按钮 ☒动态表单 ☒展开 ☒有效

访问地址：/CommonModule/Supplies/Index

说明：

确定

取消

重要栏位：

- 模块名称：该菜单模块显示的名称。
- 上级模块：该菜单模块的上一级菜单。
- 模块分类：该菜单模块是页面还是父目录。
- Icon 图标：该菜单模块显示的小图标文件名。
- 级别层次：菜单所在层级。
- 选项：该模块需要哪些扩展功能
- 访问地址：该模块的路径（非常重要）

5、配置系统按钮

配置完系统模块以后，新建的模块就可以在系统中显示，并且可以进行权限管控，而模块中还会有新增、编辑、删除等其它功能按钮，配置系统按钮就是对这些子功能进行控制。

打开“系统管理”-“智能开发”-“系统按钮”就进入了按钮设置

欢迎首页

数据字典

系统模块

系统按钮

快速开发平台

系统管理

数据字典

智能开发

系统模块

系统按钮

系统视图

系统表单

接口管理

数据库管理

报表管理

系统日志

单据编码

导入配置

基础资料

公司管理

部门管理

角色管理

岗位管理

用户组管理

用户管理

供应商管理

按钮列表 - 供应商管理

刷新

新增

编辑

删除

详细

复制

粘贴

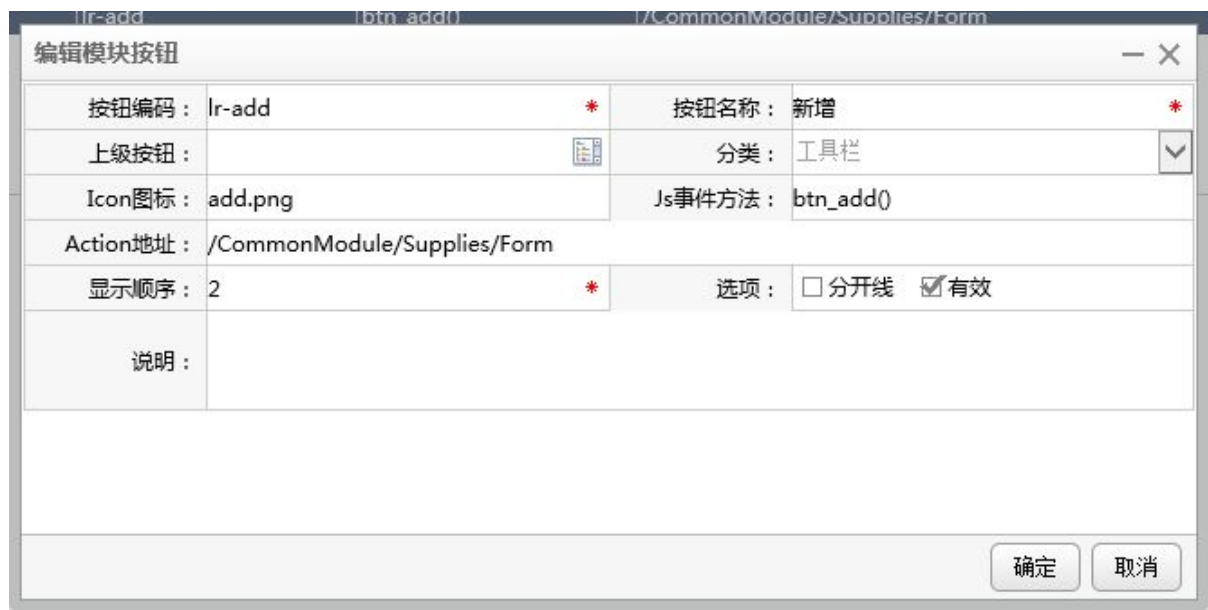
离开

工具栏

右击栏

按钮	编码	Js事件	Action地址	分开线	有效
1 刷新	lr-refresh	windowload()		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
2 新增	lr-add	btn_add()	/CommonModule/Supplies/Form	<input type="checkbox"/>	<input checked="" type="checkbox"/>
3 编辑	lr-edit	btn_edit()	/CommonModule/Supplies/Form	<input type="checkbox"/>	<input checked="" type="checkbox"/>
4 删除	lr-delete	btn_delete()	/CommonModule/Supplies/Delete	<input type="checkbox"/>	<input checked="" type="checkbox"/>
5 明细	lr-detail	btn_detail()	/CommonModule/Supplies/Detail	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
6 离开	lr-exit			<input type="checkbox"/>	<input checked="" type="checkbox"/>

选中“供应商管理”点击工具栏上的新增就可以为模块增加要显示的按钮。



编辑模块按钮	
按钮编码: lr-add *	按钮名称: 新增 *
上级按钮:	分类: 工具栏
Icon图标: add.png	Js事件方法: btn_add()
Action地址: /CommonModule/Supplies/Form	
显示顺序: 2 *	选项: <input type="checkbox"/> 分开线 <input checked="" type="checkbox"/> 有效
说明:	
<div>确定</div> <div>取消</div>	

重点栏位:

按钮名称: 按钮在界面上显示的名称。

上级按钮: 如果该按钮是要在点击上级按钮后才显示, 这里就需要选择上级按钮, 一般不需要。

Icon 图标: 按钮显示的小图标文件名。

JS 事件方法: 点击按钮要调用的 JS 方法 (记得方法名要带括号, 跟页面上的方法名保持一致)。

Action 地址: 如果该按钮会打开新的窗口或者访问其它页面这里就要填写, 否则服务端会认为是恶意请求而将其过滤掉。

分开线: 按钮后带一根竖线, 一般用于把不同功能的按钮隔开。

6、功能授权

系统里所有的功能都是有权限管控的, 所以开发完成以后要进行授权, 详细方法请参见使用手册。



这样我们编译过后就可以看到刚刚开发出来的“供应商管理”功能了。



7、代码微调

FDMS 为了充分保证开发的灵活性，所有代码都是公开出来的，用户可以自行调整源代码，以下举几个例子让用户熟悉编码风格。

7.1、绑定动态数据源的下拉框

在 FDMS 中一般有两种常用的下拉框。

第一种是绑定数据字典的。在数据字典中定义的项目可以通过一个 js 函数绑定到下拉框。



代码: `BindDropItem("#AccountsMethod", "accountmethod", "=="请选择=="");` //绑定结算方式

第一个参数是要绑定的下拉框控件 ID

第二个参数是数据字典中数据项的分类编码



在数据字典分类目录里选中“结算方式”在右侧就可以将这个分类编码复制下来。

第三个参数是下拉框的默认值。

第二种下拉框是直接绑定其它数据源的，比如说绑定：用户、客户、供应商等，在框架中多处用到这些下拉框，使用者可以直接将 JS 代码复制过来，更改一下绑定的控件即可。

//负责人自动补全

```
function UserAutocomplete() {
    var $LeadingOfficialId = $("#LeadingOfficialId");
    $LeadingOfficialId.bind("keyup", function (e) {
        if (e.which != 13 && e.which != 40 && e.which != 38) {
            DataSource();
        }
    }).focus(function () {
        $(this).select();
        DataSource();
    });
}
```

```

});
//上, 下键盘回调
autocompletekeydown("LeadingOfficialId", function (data) {
    $("#LeadingOfficialId").val(data.RealName)
});
//获取数据源
function DataSource() {
    AjaxJson("/CommonModule/User/Autocomplete", { keywords: $LeadingOfficialId.val() },
function (DataJson) {
    var html = "";
    $.each(DataJson, function (i) {
        html += "<tr>";
        html += '<td id="UserId" style="display: none;">' + DataJson[i].userid +
'</td>';
        html += '<td id="Code" style="width: 60px;">' + DataJson[i].code + '</td>';
        html += '<td id="RealName" style="width: 80px;">' + DataJson[i].realname +
'</td>';
        html += '<td id="DepartmentName">' + DataJson[i].departmentname + '</td>';
        html += "</tr>";
    });
    //点击事件回调
    autocomplete("LeadingOfficialId", $LeadingOfficialId.width() + "px", "200px",
html, function (data) {
        $("#LeadingOfficialId").val(data.RealName)
    });
});
}
}

```

原理非常简单，都是通过 ajax 拿到数据源后遍历行画表格把下拉框绘制出来，用两次一般都能自己写得非常灵活，照搬系统中现有的这些下拉框也足够平时使用。

7.2、LR-FDMS 系统常用 JS 库的使用

FDMS 系统已经把一些经常用的 js 函数集成进来放在 learunui-framework.js 中，在母版页中已经引用，用户可以直接在页面上调用，每个函数都有详细注释，用户可以参照注释使用。

7.3、重点代码解读

7.3.1、表单提交代码

```

//保存事件
function AcceptClick() {
    if (!CheckDataValid('#form1')) {
        return false;
    }
    var postData = GetWebControls("#form1");
    AjaxJson("/CommonModule/Supplies/SubmitForm?KeyValue=" + KeyValue, postData, function
(data) {
        tipDialog(data.Message, 3, data.Code);
        top.frames[tabiframeId()].windowload();
    });
}

```

```

        closeDialog();
    });
}

```

以上是 LR-FDMS 中最常见的表单提交方式

`var postData = GetWebControls("#form1")` 函数能将指定表单中的内容转换为 json, 然后通过 ajax 方式提交数据。

`AjaxJson("/CommonModule/Supplies/SubmitForm?KeyValue=" + KeyValue, postData, function (data)`
这里 SubmitForm 是继承了控制器基类中的, 它在接到 ajax 请求后把 json 数据转换成实体, 然后插入数据库。

这里要注意了, 前台视图中的控件名一定要与数据库、实体中的字段名一致, 否则将无法提交数据

7.3.2、记录日志

当我们在程序中遇到异常需要记录时可以通过以下代码记录

```
Base_SysLogBll.Instance.WriteLog("", OperationType.Query, "-1", "异常错误: " + ex.Message);
```

7.3.3、动态表单

LR-FDMS 中可以定义动态表单, 定义了动态表单的页面需要在页面中加入以下这段代码, 详情请参见用户管理的表单模块。

```

<!--自定义属性-->
<div id="CustomAttribute" class="tabPanel" style="display: none;">
    @Html.Partial("_BuildForm")
</div>

```

7.3.4、一个表单中多 tab 处理



为了界面排版更加美观内容比较多的页面可以分出几个 tab, 加一个 tab 需要在页面中加一个 div, 如用户管理中的基本信息, 记得选好 tab 的样式 tabPanel:

```

<!--基本信息-->
<div id="basic" class="tabPanel">

```