# 基本概念

## 模块化

### 好处

如果按照统一规范编写代码，模块化的好处简单说来就是可以更方便地使用别人的代码，想要什么功能，就加载什么模块。

### 初步了解

Web应用的快速发展、后端逻辑的前端化，在客观上要求多人协作编写js代码，这样就带来了以下问题：

* 全局变量命名混乱冲突导致系统错误
* 代码集中少量文件导致开发效率降低
* 容易出现功能重复开发

模块化框架对于按需加载、模块划分、代码复用、自动文档、单元测试、团队合作等都有很大好处。

模块就是实现特定功能的一组方法，理想情况下，开发者只需要实现核心的业务逻辑，其他都可以加载别人已经写好的模块。

表格 1 模块化

|  |  |
| --- | --- |
| 实现方式 | 优缺点 |
|  | 优点：代码简单啊  缺点：全是全局变量，容易冲突 |
|  | 优点：属性封装在对象中，看起来是减少了上述问题  缺点：会暴露所有模块成员，内部状态可以被外部改写（module1.\_count = 5） |
|  | 立即执行的函数写法  优点：隐藏了私有变量，只返回需要暴露的内容  这是模块化基本的实现思路 |
|  | 放大模式，如果一个模块很大，必须分成几个部分，或者一个模块需要继承另一个模块时使用  优点：对基本方式的加强，可对特定的对象进行增强，如对左边的module1添加一个方法 |
|  | 宽放大模式（与"放大模式"相比，区别在于"立即执行函数"的参数可以是空对象）  优点：避免“放大模式”中因加载顺序导致的未定义的错误 |

### 规范

1. CommonJS

在CommonJS中，有一个全局性方法require()，用于加载模块。

CommonJS存在的问题在于如果目标模块很大，则需要全部等待加载完之后代码才可以继续执行，这样表面看起来是假死状态。

1. AMD (Asynchronous Module Definition)

AMD也采用require()语句加载模块，不同的是它是异步的

require([module], callback);

等各模块加载完成后调用callback回调函数

require.js实现了AMD规范

## Echarts

ECharts，纯Javascript图表库，基于Canvas，底层依赖ZRender，商业产品常用图表库，提供直观，生动，可交互，可个性化定制的数据可视化图表。创新的拖拽重计算、数据视图、值域漫游等特性大大增强了用户体验，赋予了用户对数据进行挖掘、整合的能力。图表类型支持折线图（区域图）、柱状图（条状图）、散点图（气泡图）、K线图、饼图（环形图）、雷达图（填充雷达图）、和弦图、力导向布局图、地图，同时支持任意维度的堆积和多图表混合展现。

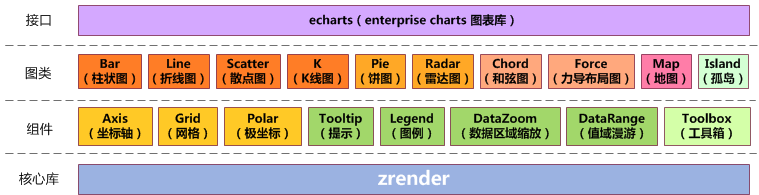


图 1 Echarts 图表类型

根据实际的业务需求选择适合的图标类型并引入相关的模块。

# 例子

## 效果图

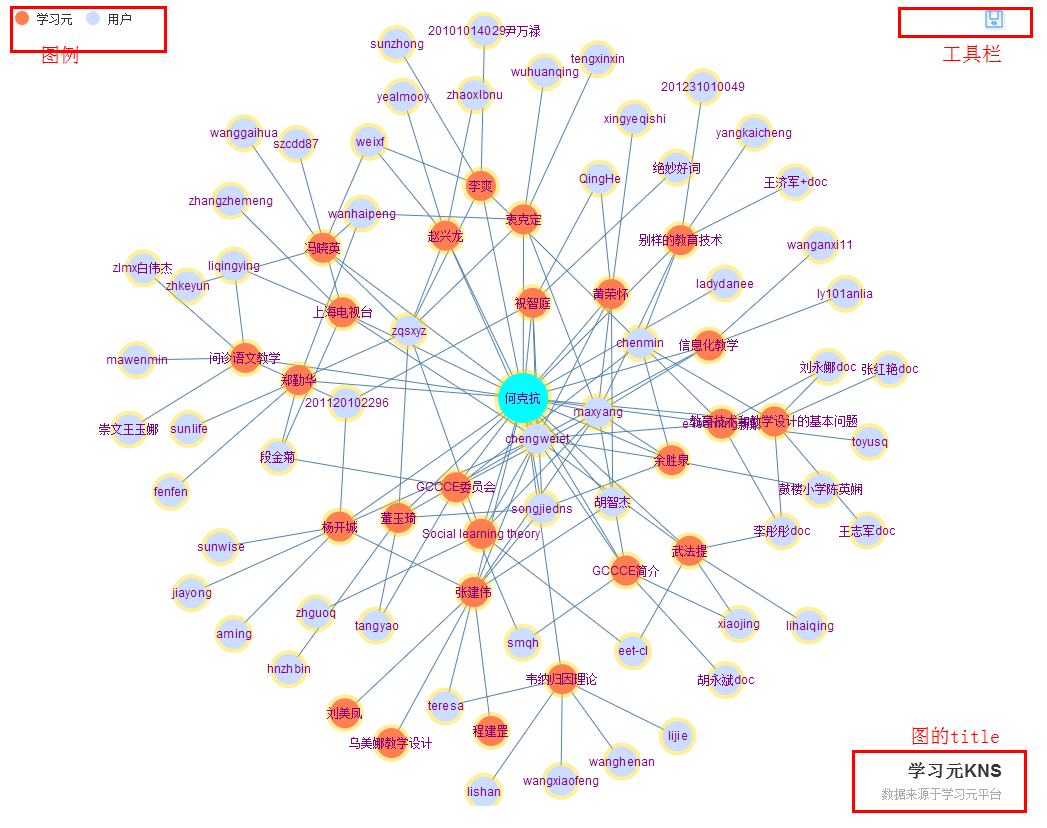


图 2 效果图

本例子使用了力导向布局图。

## 实现

### 准备工作

空html页面

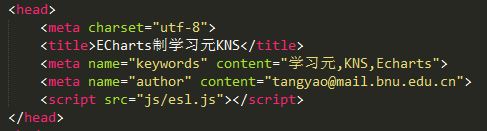
<body>标签中包含一个有显示大小的div容器

<div id="main" style="height: 800px;width:1000px;border: 1px solid #FFF; padding: 10px; margin:0 auto"></div>

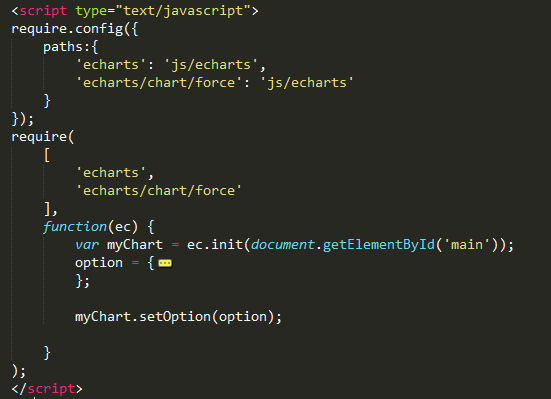
### 模块加载

1. 模块加载器esl

esl.js是一个模块加载器，在头部引入可用于之后加载需要的相应模块



1. 配置Echarts需要的模块（本例子中使用力导向布局图）



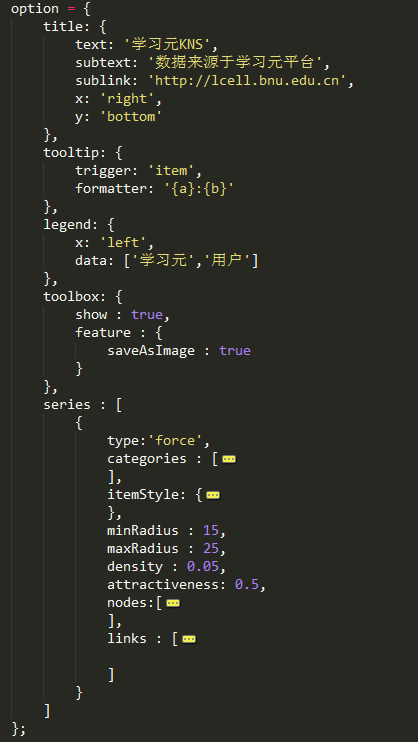
require.config配置好后就可以通过动态加载使用echarts。

在path中根据echarts.js文件存放的位置配置路径（本例中使用了力导向图force）

在require函数的回调函数中初始化echarts（通过调用init函数）

Option对象可设置图标的样式、数据源等

### 配置Option对象



title——图标的标题，可设定文字、副标题、副标题的URL以及标题显示的位置

tooltip——鼠标移过的时候显示的提示框的内容（如图 3）。其中formatter是内容格式器，不同的图标回调得到的显示值不同，本例中是显示系列值和项的名称。trigger表示触发类型，item表示数据触发，可选的还有axis（一般用于有坐标轴的条形图等）



图 3提示框

legend——图例。单击图例可隐藏或显示该系列的值（图 4）

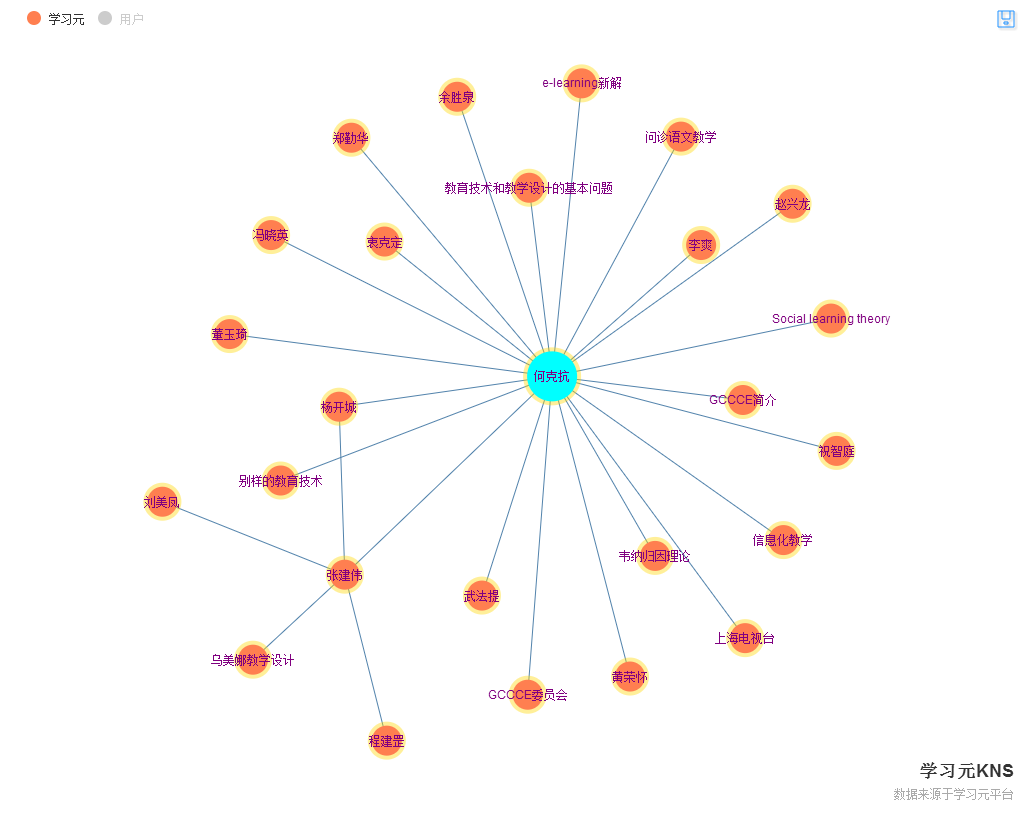


图 4 图例控制显示

toolbox——每个图表最多有一个工具箱。通过配置toolbox的feature属性可开启以下功能：工具箱提供辅助线标志、框选区域标缩放、图表类型切换功能、数据视图、还原以及保存为图片功能。本例中仅使用了保存为图片。

Series——驱动图表生成的数据内容数组，数组中每一项为一个系列的选项及数据。本例中仅有一个系列。



图 5 力导向图配置

myChart.setOption(option); //为图表实例设置配置项生效。