[{

name:”jack”,

age:22,

birth:”1994-10-10”

}]

**BQL**

BQL(Bind Query Library)是基于JQL(Json Query Library)，在其基础上添加了对html元素的绑定，可以以类SQL的方式对Json数据或是其他类型数据进行查询等操作，并绑定到UI上。

1. **Select(attr,run)**

选择属性

1. **select(“name”)**
2. **Select(“distinct name”)**
3. **update(attr,value,run)**

更新属性

**A.update(“”)**

1. **Add(attr,value,run)**

添加属性

1. **Add(“height”,”175”)**
2. **Add({height:175})**
3. **Remove(attr,run)**

删除属性

**A.Remove(“age”)**

1. **Insert(attr,value,index,run)**

插入元素

**a.Insert(“name,age,birth”,”bob,23,1995-10-10”)**

**b.Insert({name:’bob’,age:23,birth:’1995-10-10’})**

**c.Insert(“name,age,birth”,”bob,23,1995-10-10”,1)**

1. **Delete(index,run)**

删除元素

1. **delete()**

删除元素，与where一同使用，若是没有where，则将删除所有元素

1. **Delete(0)**

删除索引等于index值的元素

1. **Where(attr,value,run)**

Where判断

**a.where(“name”,”jack”)**

**b.Where({name:’jack’})**

1. **Where(“name==’jack’”) where(“$index==0”) where(“$i==0”)**

布尔表达式，完全遵循js语法，$index或$i 为当前元素在整个array中的位置

可以使用正则表达式实现like功能

1. **groupBy(attr,run)**

聚合，与select聚合函数（count,sum,avg,min,max,first,last,count(distinct)）一起使用，也可单独使用

**a.groupBy(“name”)**

1. **orderBy(attr,order,type,run)**

排序

1. **orderBy(“name”)**
2. **orderBy(“name”,”desc”)**
3. **orderBy(“birth”,”desc”,”date”)**
4. **orderBy(“birth”,”date”)**
5. **Run()**

执行查询，也可在上述方法参数列表最后添加一个true来执行,如果数据有改动，刷新页面相关的dom元素

1. **Set(data,run)**

用一个数据替换BQL对象中的数据，也就是更新所有数据

1. **Get()**

获取所有数据的克隆副本，对结果做修改不会影响到本体

1. **Data()**

获取所有数据，对结果做修改会影响到本体

1. **Clear()**

将BQL对象中的数据置为null

1. **Init(data)**

初始化BQL对象

1. **BQL.init(element)**

初始化一个或全部dom元素（包含b-bind或b-loop属性的），并生成一个以该属性值为名的全局的BQL对象，所有方法都在该对象上执行，页面加载是会执行BQL.init()，若动态添加 bql dom元素，请使用 BQL.init(element) 再使用BQL对象

属性

1. b-bind 单值
2. B-loop 数组
3. B-update input值改变 是否更新数据 默认为false oninput 更新数据
4. B-refresh 是否onchange 刷新页面 默认为false
5. B-callback 刷新ui的回掉函数 this(或obj)指代的就是当前的dom元素 data是数据
6. B-init 初始化的数据
7. B-each 下文中引用的变量名 默认为each 只能与 b-loop同时使用才有效

**JQL**

BQL是基于JQL，JQL脱离了UI与绑定，是对数据直接做操作，用法与BQL类似，不过其中没有init方法，若要对data使用JQL，可以这样使用 Jql(data).select(“\*”).run()。

JQL可以以类SQL的方式对Json数据或是其他类型数据进行查询等操作。

1. **Select**
2. **Update**
3. **Add**
4. **Remove**
5. **Insert**
6. **Delete**
7. **Where**
8. **groupBy**
9. **orderBy**
10. **Run**
11. **Set**
12. **Get**
13. **Data**
14. **Clear**
15. **Jql**