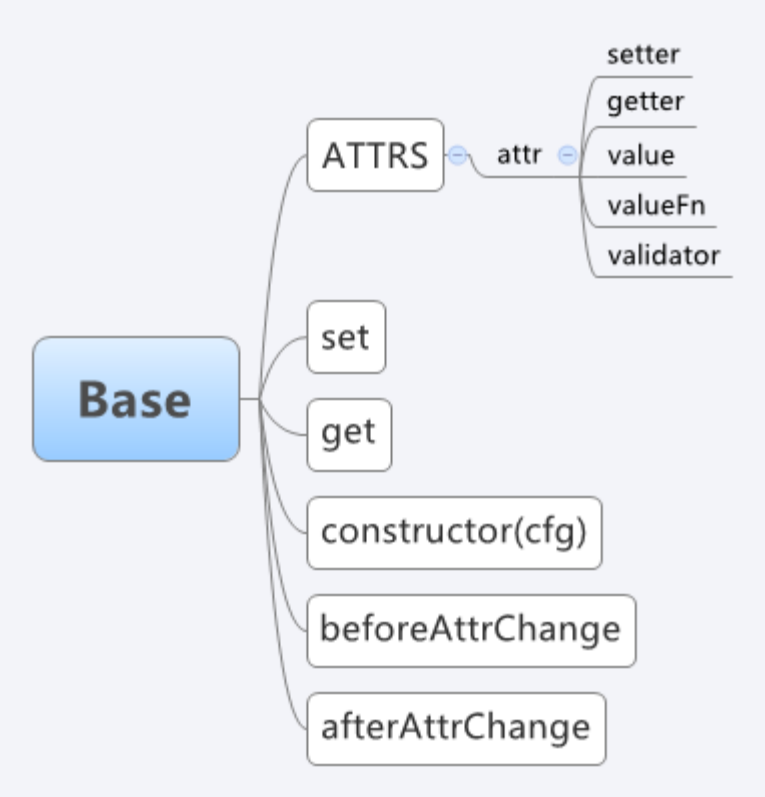
技术嘉年华 2011
Taobao Open Developer Conference

KISSY Component

yiminghe@gmail.com

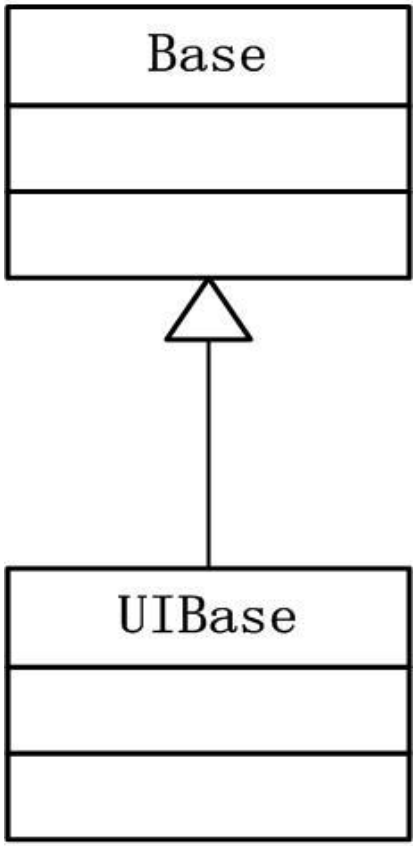
base


Base
ATTRS
attr
setter
getter
value
valueFn
validator
set
get
constructor(cfg)
beforeAttrChange
afterAttrChange

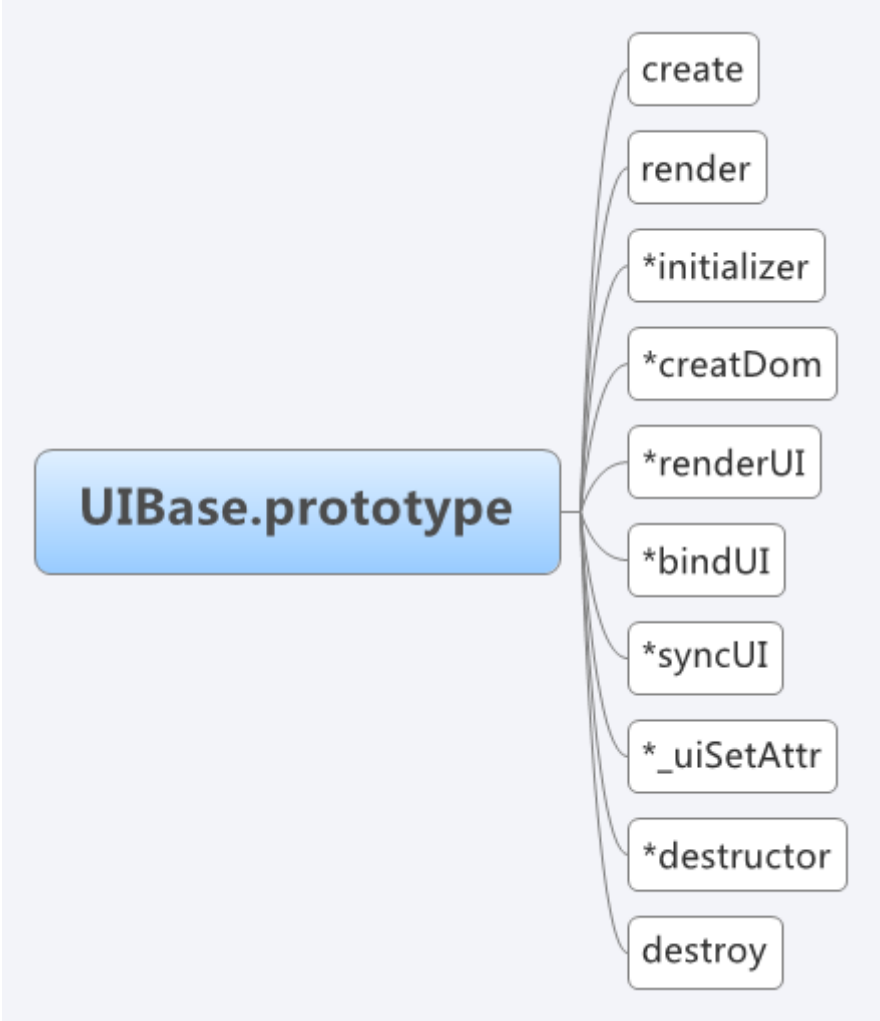
示例

```
function X(){
}
X.ATTRS={
    y:{
        value:1,
        setter:function(){
        },
        getter:function(){
        }
    }
};
S.extend(X,S.Base);

var x=new X({
    y:3
});

x.on("beforeYChange",function(e){
    e.newVal;
    e.prevVal;
});

x.on("afterYChange",function(e){
    e.newVal;
    e.prevVal;
});

x.set("y",5);

x.get("y");
```

UIBase



UIBase.prototype



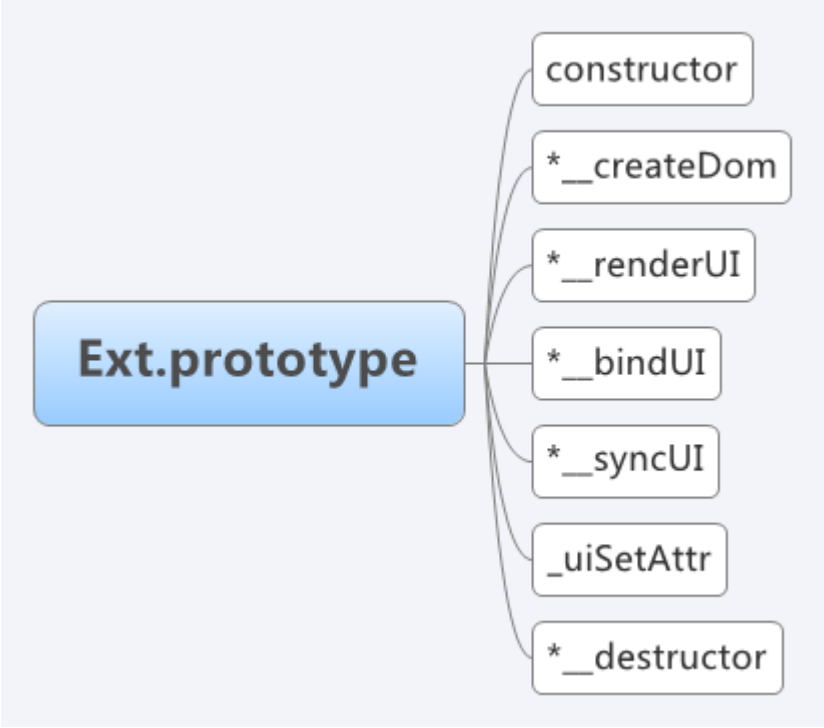
UIBase

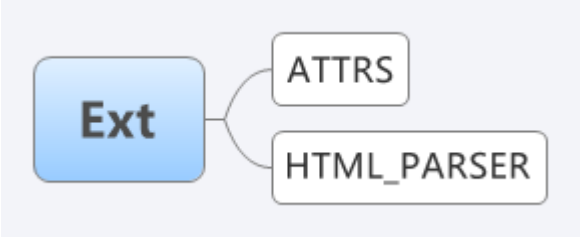
UIBase.create

- 将多个扩展和一个主类合并为一个新类
  - 考虑扩展声明周期

```
UIBase.create([UIBase.Box],{
    initializer:function(){},
    destructor:function(){},
    renderUI:function(){},
    createDom:function(){},
    bindUI:function(){},
    syncUI:function(){},
    _uiSetY:function(){}
},{
    ATTRS:{
        y:{}
    }
});
```

扩展类


Ext.prototype
constructor
*__createDom
*__renderUI
*__bindUI
*__syncUI
_uiSetAttr
*__destructor


Ext
ATTRS
HTML_PARSER

示例

```
function Box(){}

Box.ATTRS={
    width:{},
    // 渲染容器节点
    render:{}
    // 或用于定位的节点
    elBefore:{}
};

Box.HTML_PASER={
    el:function(node){
        return node;
    }
};

Box.prototpe={
    __createDom:function(){
        if(!this.get("el")){
            // create
        }
    }

    __renderUI:function(){
        // append el to dom
    },

    _uiSetWidth:function(){},

    __destructor:function(){
        this.get("el").remove();
    }
};
```

initializer

- 初始化

destructor

- 析构

_uiSetAttr

- 属性通知
  - set(“attr”) => _uiSetAttr

createDom

- 建立节点

renderUI

- 添加节点到document

bindUI

- 注册事件

syncUI

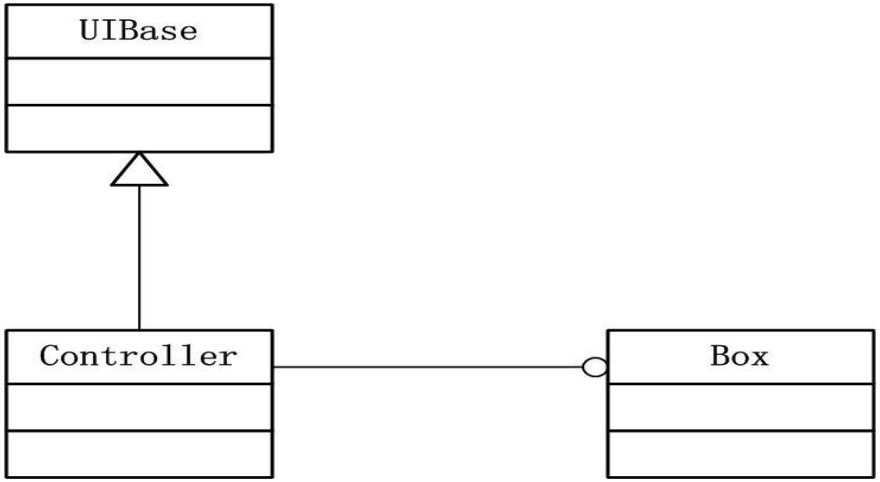
- 同步属性状态

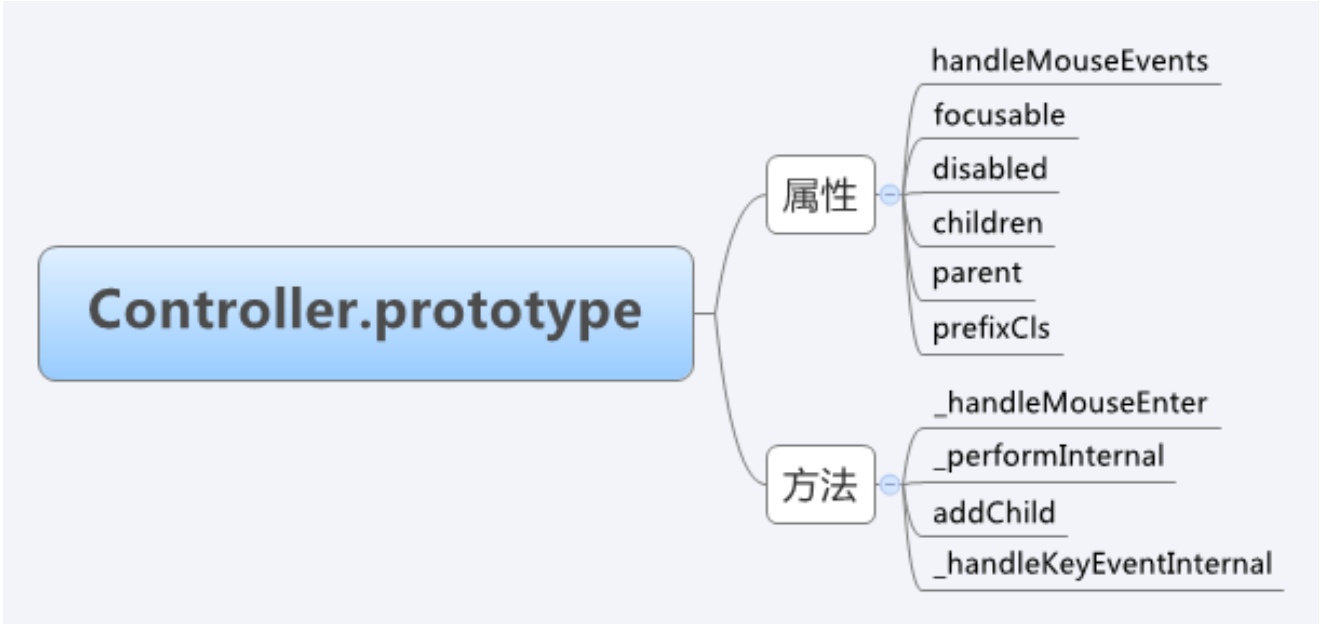
生命周期顺序

- initializer/renderUI/bindUI/sycUI/createDom
  - 父类，子类扩展类，子类
- destructor
  - 子类，子类扩展类，父类

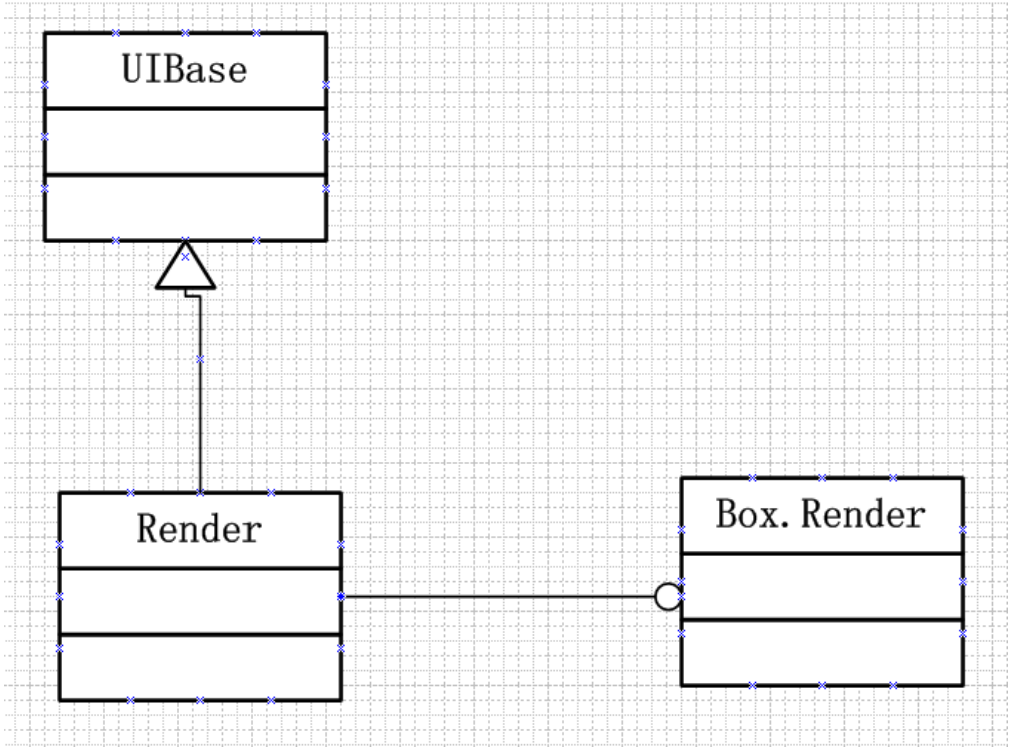
Component

- Controller
  - 事件注册
    - 鼠标，键盘
  - 组件层次
    - parent，children
  - 皮肤支持
    - prefixCls
  - mvc
    - render


UIBase
Controller
Box


Controller.prototype
属性
handleMouseEvents
focusable
disabled
children
parent
prefixCls
方法
_handleMouseEnter
_performInternal
addChild
_handleKeyEventInternal

- Render



- Controller -> Render
  - DEFAULT_RENDER

扩展

- DelegateChildren
  - 事件代理
- DecorateChildren
  - 从已有 html 层次初始化

管理

- uistore
  - css 与组件 controller 的 map 中央管理
  - 用于从 html 生成对应组件

```
var Menu=UIBase.create(Controller,[DelegateChildren,DecorateChildren],{
    _handleKeyEventInternal:function(){
        // down, up, then update activeItem
    }
},{
    ATTRS:{
        focusable:{
            value:true
        },
        handleMouseEvents:{
            value:true
        },
        activeItem:{
            view:true
        }
    }
});

UIStore.set("menu",Menu);

var MenuRender=UIBase.create(Render,[],{
    _uiSetActiveItem:function(item){
        el.attr("aria-activedescendant",item.id);
    }
});
var MenuItem = UIBase.create(Controller,[],{
    _performInternal:function(){
        // 当前项点击
    },
    _handleMouseEnter:function(){
        // 设置高亮状态
    }
},{
    ATTRS:{
        focusable:{
            value:false
        },
        handleMouseEvents:{
            value:false
        }
    }
});

UIStore.set("menuitem",MenuItem);
```

使用

- 完全生成
- 从 html 生成

```
var menu=new Menu({
    render:"#container"
});
menu.addChild(new Menu.Item());
menu.render();

<div class='ks-menu' id='m'>
    <div class='ks-menuitem'>item1</div>
</div>

var menu=new Menu({
    srcNode:'#m'
});
menu.render();
```

技术嘉年华 2011
Taobao Open Developer Conference

组合组件编写

- New AutoComplete