UUI开发手册1.0

上海思眭信息技术有限公司

2014-07-31

UUI技术团队

目 录

[1 引言 3](#_Toc394665938)

[2 理念 3](#_Toc394665939)

[3 发展时间线——代表作 4](#_Toc394665940)

[4 体系 5](#_Toc394665941)

[5 SDK架构 6](#_Toc394665942)

[6 编写规范 6](#_Toc394665943)

[6.1 要素 6](#_Toc394665944)

[6.2 Module 7](#_Toc394665945)

[6.2.1 Module定义 7](#_Toc394665946)

[6.2.2 Module使用 8](#_Toc394665947)

[6.3 命名 9](#_Toc394665948)

[6.4 页面模板和Widget名 10](#_Toc394665949)

[7 方法速查 12](#_Toc394665950)

[(1): uui/jquery.min.js 12](#_Toc394665951)

[(2): uui/uui-loader.js 12](#_Toc394665952)

[(3): /uui/widget/jqui/ui/i18n/jquery-ui-i18n.js 14](#_Toc394665953)

[(4): /uui/widget/jqui/ui/jquery-ui.js 14](#_Toc394665954)

[(5): /uui/util/dateutil.js 14](#_Toc394665955)

[(6):/uui/util/objectutil.js 15](#_Toc394665956)

[(7):/uui/util/stringutil.js 15](#_Toc394665957)

[(8):/uui/core/uclazz.js 16](#_Toc394665958)

[(9):/uui/core/ucomp.js 17](#_Toc394665959)

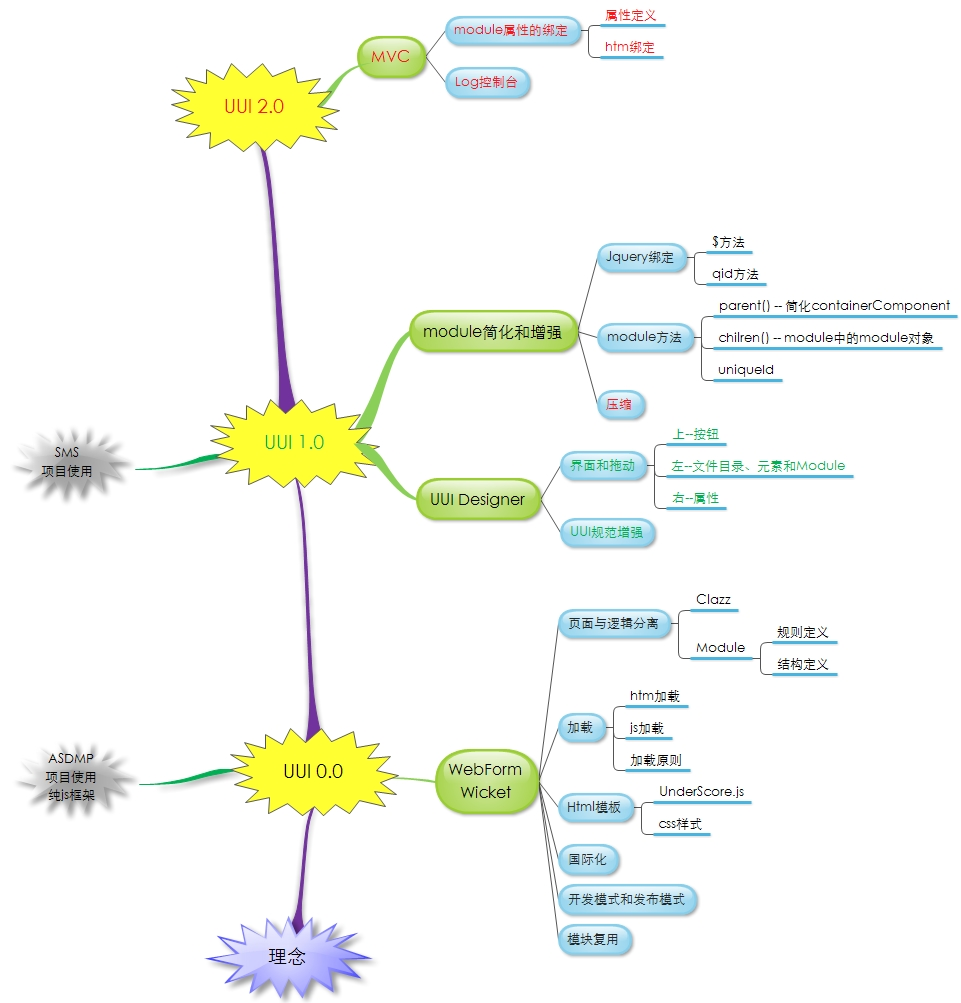
# 引言

前端架构很多，适合自己的就是最好的。

# 理念

* 工业化
* 简洁明了

# 发展时间线——代表作



# 体系



* SDK
* 设计器（包含日志控制台）
* Module测试工具
* 压缩打包工具

# SDK架构



# 编写规范

# 要素

* 无页面Module的主加载页（举例：index.html）

head里必须有：

<script type="text/javascript" src="uui/jquery.min.js"></script>

<script type="text/javascript" src="uui/uui-loader.js"></script>

启动时加载基本的：

* + $.u.config.appname是发布的路径
  + $.u.inituui是初始化整个UUI环境
  + $.u.load是加载自己定义的Module（实际类似类的import）
  + 最后new个实例出来（记得带id）

$(function () {

//$.u.config.appname = 'asdmp/uui/';

$.u.config.appname = '';

// 环境初始化工作

$.u.inituui();

$.u.load('com.asdmp.layout');

new com.asdmp.layout();

});

* 有页面Module的主加载页（举例：index.html）

页面片段：

<div umid="layout" umodule="com.asdmp.layout"></div>

JS片段：

$(function () {

//$.u.config.appname = 'asdmp/uui/';

$.u.config.appname = '';

// 环境初始化工作

$.u.inituui();

$.um($("div[umid='layout']"))

});

* Module

Module最多具备三个文件：名称.htm（展现）、名称.js（逻辑）、名称-i18n-语言.js（国际化），如下例子。语言的配置在$.u.config.region属性，默认“zh”。



三个文件不是必须，通过定义Module参数{ usehtm: true, usesuperhtm: true, usei18n: false }来控制。

# Module

# Module定义

//@ sourceURL=uuitest.comp.compbasic.comp.comp2

$.u.define('uuitest.comp.compbasic.comp.comp2', null, {

init: function () {

this.comp2display = 'hello comp2';

this.jqdivid = null;

},

afterrender: function () {

this.jqdivid = this.qid('comp2id')

console.log(this.jqdivid);

}

}, { usehtm: true, usei18n: true });

uuitest.comp.compbasic.comp.comp2.widgetjs = ['../js/js1.js'];

uuitest.comp.compbasic.comp.comp2.widgetcss = [{ path: '../css/css1.css' }];

第一行注释用于谷歌浏览器能够加载。

* $.u.define用于定义Module
  + 第一个参数是Module的名称，**名称实际也是对应的目录路径**（参考java），比如：'uuitest.comp.compbasic.comp.comp2'，对应目录就是$.u.config.appname + “uuitest/comp/compbasic/comp”下的”comp2.js”与“comp2.htm“。
  + 第二个参数是继承的Module名。
  + 第三个参数就是**内部方法**的覆盖。Component提供如下内部方法：
    - init初始化，**声明**变量名称
    - beforechildrenrender，在子module加载之前的一些操作
    - afterrender，htm加载和渲染完成，常用的就是这个
* widgetjs和widgetcss是类的常量，用来加载类中使用的jq-widget，当然，这个可以不用，直接写在Modulehtm的<head>标签中，**建议优先写在js里**。

# Module使用

$.u.load('uuitest.comp.compbasic.comp.comp2');

var cClsObj = new uuitest.comp.compbasic.comp.comp2($("div[umid='layout']"),"4");

cClsObj.override({

bark2 : function () { return("barkig...") }

});

* $.u.load用于加载Module
  + 方法只有一个参数，类似java的import方法，也可以是字符串数组，一次加载多个组件
  + 新建的Module实例，有自动生成的”\_id”属性标识唯一，还有“cClsObj.$”方法获取当前组件的jQuery选择器。
* 对象的override

Override是对象方法的覆盖，主要用于对象之间的交互。

$.u.define('comp.b', null, {

on\_xx\_widget\_click : function (){} //ModuleB暴露的方法，B的请求

});

var bx = new com.b();

bx.override({

on\_xx\_widget\_click: function (){currentObject.c. show\_xx \_widget();}//覆盖Moduleb暴露的方法

});

# Module的子Module

有三个Module，ModuleA、B、C，B和C都是在A中new出来

有三个Module，ModuleA、B、C，B和C都是在A中new出来

# 命名

* Module的命名

以com.xx.xx这类形式组织，目录编排com/xx，该目录下最少有xx.js，最多有xx.js、xx.htm、xx-i18n-语言.js。如果是容器Module（包含其他Module的Module），容器内定义的Module为com.xx.xx.xx（更进一级）。

* Module实例

Module实例中第一个参数是唯一标识名。

实际，Module的 this.id 就是以类名+唯一标识名参数构成，比如com.xx.xx，名称为id1，自动构成的this.id为com-xx-xx-id1-

* impl方法中，事件最好以“**call \_内部Module/Widget名\_事件**”规范编写。

# 页面模板和Widget名

<div id="{@ o.id @}comp2id">

<ul>

<li>ID：{@ o.id @}</li>

<li>类名：{@c.clazzname@}</li>

<li>父类名：{@c.superclazz.clazzname@}</li>

<li>全局版本：{@g.i18n.version @}</li>

<li>类的i18n：{@c.i18n.test@}</li>

<li>对象的\_super之前的属性：{@o.comp2display@}</li>

<li>对象的\_super之后的属性：{@o.notdisplay@}</li>

<li>对象的方法：{@o.bark2()@}</li>

<li><a href="../htmlfortest.html">xxxxxx</a></li>

<li><img src="../testpng.png" /></li>

</ul>

</div>

* 页面模板中宏定义有如上以{@@}括在里面的内容。这些内容在执行过程中会被转义，有三种基础对象：o（对象自身）、c（Module自身）、g（全局）
* id="{@ o.id @}comp2id"，comp2id即Widget名，如果有impl的click方法，那么事件命名为on\_comp2id\_click。
* Jq获取comp2idModule方法为： this.comp2id = $('#' + this.id + 'comp2id')
* 所有<a>标签和<img>标签中的链接地址（除了“#”），都是相对当前htm页面的地址，这些地址会在加载页面模板的时候重新计算。

# 方法速查

## (1): uui/jquery.min.js

## (2): uui/uui-loader.js

|  |  |  |  |
| --- | --- | --- | --- |
| 定义全局变量$.u  初始化全局变量$.u | $.u = $.u || {} | | |
| $.u.config | region | 语言,默认为’zh’ |
| I18n | 国际化定义,默认为{} |
| appname | Web工程名称,默认为’’ |
| model | dev开发模式,按corejs、corecss加载;dep发布模式,一次性加载（前提先加载jquery）  默认为’dev’ |
| $.u.\_global | domain | Web地址全路径  例:http://ip:port/appname |
| protocol | 协议名称  例:’http:’ |
| scripturls | 默认为[] |
| devcorejs | 基础js路径 |
| devcorecss | 基础css路径  id: 'jqui'  path: 'xxx.css'  disabled : false  什么要id,是因为 |
| depcorejs | 扩展用默认为[]  一般是没有的,直接index.htm页面引入,留着以后做规范 |
| depcorecss | 扩展用默认为[]  一般也是没有的,直接index.htm页面引入,留着以后做规范,里面对象例子{ id: 'jqui', path: '/uui/widget/jqui/themes/base/jquery-ui.css' , disabled : false} |
| $.u.\_absolutizeuri | function remove  DotSegments  (input) | 解析URL方法 |
| function parseURI(url) | 解析URL方法 |
| Result | 返回加载的css,js,图片等 |
| $.u.inituui | 给domain赋值:web全路径  给protocol:协议  过滤css,js,图片避免重复加载  加载国际化语言  /uui/widget/jqui/ui/i18n/jquery-ui-i18n.js  加载corejs包括:  /uui/widget/jqui/ui/jquery-ui.js  /uui/util/dateutil.js  /uui/util/objectutil.js  /uui/util/stringutil.js  /uui/core/uclazz.js  /uui/core/ucomp.js  加载corecss 包括:  /uui/widget/jqui/themes/base/jquery-ui.cs由于此处  devcorecss 中设置 disabled为false因此次/jquery-ui.css不起作用 | |
| $.u.\_absurl | 方法:获取组织绝对路径  参数:  Path:绝对路径  Basepath:参考路径  返回各js,css,图片以及htm的绝对路径 | |
| $.u.loadjs | 方法:加载js  参数  Jspath: jspath为js的路径,如果jsPath  是以domain打头,则是绝对路径;  否则是相对路径.  Basepath:,basepath为jspath相对路径时的参考路径.  forceable: forceable忽略发布模式强制加载. | |
| $.u.loadcss | 方法:加载css  参数  Cssobj: cssobj为对象,如果cssobj.path是以domain打头,  则是绝对路径;否则是相对路径  Basepath: basepath为jspath相对路径时的参考路径.  Forceable: forceable忽略发布模式强制加载. | |
| $.u.loadhtm | 方法:加载htm  参数  Htmpath:htmpath为htm的路径,如果htmPath是  以domain打头,则是绝对路径;否则是相对路径。  Basepath:basepath为jspath相对路径时的参考路  径。 | |
| $.u.getjssrcfromhead | 方法:加载htm页面head中引用的js  参数:headstr | |
| $.u.getcsssrcfromhead | 方法:方法:加载htm页面head中引用的css  参数:headstr | |
| $.u.getcssrawfromhead | 方法:方法:加载htm页面head中直接编写的css  参数:headstr | |
| $.u.xmltostring | 方法:xml转string工具  参数:xmlData  返回值:xmlString | |

## (3): /uui/widget/jqui/ui/i18n/jquery-ui-i18n.js

## (4): /uui/widget/jqui/ui/jquery-ui.js

## (5): /uui/util/dateutil.js

|  |  |  |
| --- | --- | --- |
| 扩展date方法 | Date.prototype.format | 方法:对Date的扩展,将 Date 转化为指定格式的String  参数:fmt 日期格式  说明:月(M)、日(d)、12小时(h)、24小时(H)、分(m)、秒(s)、周(E)、季度(q) 可以用 1-2 个占位符  年(y)可以用 1-4 个占位符,毫秒(S)只能用 1 个占位符(是 1-3 位的数字)   例如  (new Date()).format("yyyy-MM-dd hh:mm:ss.S") ==> 2006-07-02 08:09:04.423  (new Date()).format("yyyy-MM-dd E HH:mm:ss") ==> 2009-03-10 二 20:09:04  (new Date()).format("yyyy-MM-dd EE hh:mm:ss") ==> 2009-03-10 周二 08:09:04  (new Date()).format("yyyy-MM-dd EEE hh:mm:ss") ==> 2009-03-10 星期二 08:09:04  (new Date()).format("yyyy-M-d h:m:s.S") ==> 2006-7-2 8:9:4.18 |
| Date.prototype.parsestring | 方法:格式化日期  参数:dateFmt 日期格式  isFmtWithZero:zero补充,默认有 |

## (6):/uui/util/objectutil.js

暂时不能扩展方法,影响jQuery layout

## (7):/uui/util/stringutil.js

|  |  |  |
| --- | --- | --- |
| 扩展String(var)方法  (String.prototype.) | startwith | 方法:判断是否以某个字符或字符串开始。  参数:prefix 字符  返回:true/false |
| endwith | 方法:判断是否以某个字符或字符串结束。  参数:prefix 字符  返回:true/false |
| left | 方法:返回某字符或字符串左边的字符或字符串  参数:n 某字符或字符串  返回:左边的字符串 |
| right | 方法:返回某字符或字符串右边的字符或字符串  参数:n 某字符或字符串  返回:右边的字符串 |
| trimAll | 方法:去除所有空格  参数:无  返回:字符串 |
| trim | 方法:去除两端空格  参数:无  返回:字符串 |
| ltrim | 方法:去除左端空格  参数:无  返回:字符串 |
| rtrim | 方法:去除右端空格  参数:无  返回:字符串 |
| reverse | 方法:翻转，字符按倒序排列  参数:无  返回:字符串 |
| addcharat | 方法:加字符  参数: ind-加字符的位置如3表明在第3个字符后增加 字符串  Char-需要增加的字符串  返回:字符串 |
| removecharat | 方法:去除指定位置的字符或者字符串  参数: ind-坐标  返回:去除坐标位置的字符或字符串 |
| Striptags | 方法:去除“<”和“>”  参数:无  返回:字符串 |
| Replaceall | 方法:替换所有  参数: oldstr-待替换的字符串  newstr-替换字符串  ignoreCase-是否忽略大小写 true/false  返回:字符串 |
| topropercase | 方法: 首字大写，其他小写  参数:无  返回:字符串 |
| byteslen | 方法:返回字符串长度（中文为2个字符串）  参数:无  返回:字符串 |

## (8):/uui/core/uclazz.js

使用说明:

**var** pCls = $.u.clazz.extend({

\_init: **function** (name) {

**this**.name = name;

},

\_initafter: **function** (name) {

},

returnTest: **function** () { **return** (**this**.name) }

});

// 1、测试类

**var** pClsObj = **new** pCls("1");

输入结果：

pClsObj.returnTest() == "1"

// 2、测试类的impl方法（追加方法）及原方法覆盖

pCls.impl({

returnTest: **function** () { **return**("2") },

returnTest3: **function** () { **return**("3") }

});

pClsObj = **new** pCls("1");

输入结果：

pClsObj.returnTest() == "4"

pClsObj.returnTest3() == "3"

// 3、测试类的继承

**var** cCls = pCls.extend({

returnTest4: **function** () { **return**(**this**.name) }

});

**var** cClsObj = **new** cCls("4");

输入结果：

cClsObj.returnTest4() == "4"

// 4、测试类方法的override

// 可以直接写(new cCls("4")).override({returnTest4: function () { return("5") }});

cClsObj.override({

returnTest4: **function** () { **return**("5") }

});

console.log(cClsObj.returnTest4() == "5");

// 5、override还可以这样，用JQuery

$.extend(cClsObj, {

returnTest4: **function** () { **return**("6") }

});

输入结果：

cClsObj.returnTest4() == "6"

## (9):/uui/core/ucomp.js

|  |  |  |
| --- | --- | --- |
| 初始化Modulecomponent | var component = $.u.clazz.extend | 定义Modulecomponent，继承$.u.clazz. |
| \_init(componetid, parentSelector) |
| \_afterinit (componetid, parentSelector) |
| \_loadhtm 加载html |
| beforerender: function (currentClazz, currentObject, bodystr)  用来覆盖，render之前调用，usehtm必须为true |
| render (currentClazz,currentObject,bodystr)  用来覆盖，usehtm必须为true，比如用jeasyui，\_render内容就是：  currentObject.parentSelector.append(bodystr);  $.parser.parse(currentClazz.parentSelector); |
| afterrender: function (currentClazz, currentObject, bodystr)  用来覆盖，usehtm必须为true，render之后调用 |
| 定义类：  $.u.definecomponent (  currentclz,: 定义的类  superclz, : 父类  currentclzmethods, : 方法  currentclzconfigexts: 扩展或覆盖类的config  ) |
|  | 加载类、类的国际化、类的css和js  $.u.loadcomponent(compClzName)  参数  compClzName：类名 |
|  | $.u.settimeout (component, strMethod, msinterval)  参数  component:  strMethod:  msinterval: |
|  | $.u.setinterval (component, strMethod, msinterval)  参数  component:  strMethod:  msinterval: |
|  | 用’-’替换分隔符  如xx.xx.xx->xx-xx-xx-id-  $.u.\_buildcomponentid (compClzName, id)  参数  compClzName：类全名  id：Moduleid |
|  | 用’/’替换分隔符  如xx.xx.xx->$.u.\_global.domain/xx/xx/xx  $.u.\_ buildcomponentpathprefix (compClzName)  参数  compClzName：类全名 |