

Table of Contents

概述	1.1
快速开始	1.2
模块	1.3
navigator(导航控制器)	1.3.1
notify(通知)	1.3.2
photo(相机/相册)	1.3.3
static(静态存储)	1.3.4
pref(固定存储)	1.3.5
fpicker(picker)	1.3.6
progress(无状态进度框)	1.3.7
qr(二维码扫描)	1.3.8
addressBook(通讯录)	1.3.9
net(网络访问)	1.3.10
组件	1.4
looper(轮播器)	1.4.1
host(多页组件)	1.4.2
web(扩展)	1.4.3
floating(小菊花进度指示器)	1.4.4
image(扩展)	1.4.5
drawerlayout(抽屉)	1.4.6
root写法	1.5

weexplus

一个用于快速生成weex项目的脚手架。

快速开始

Windows系统的用户建议使用 `Cmder` 或者 `git bash` 作为命令行工具

1. 安装 `Git`，已安装的请忽略

最简单的安装方法就是直接安装[官网的安装包](#)，至于其它的使用包管理或者源码安装的方法请自行搜索。

2. 安装 `NodeJs`，已安装的请忽略

最简单的安装方法就是直接安装[官网的安装包](#)，至于其它的使用包管理或者源码安装的方法请自行搜索。

3. 安装 `cnpm` 和 `weex-toolkit`，已安装的请忽略

使用 `cnpm` 是为了避免国内用户下载和安装 `node` 模块过慢的情况。使用命令行执行

```
npm i -g cnpm  
cnpm install -g weex-toolkit
```

4. 安装脚手架工具 `wxp`，已安装的请忽略

`wxp` 意即 `weexplus`，使用命令行执行

```
cnpm i -g wp
```

`wxp` 工具为我们提供了很多常用的开发功能，后续会一一介绍。

5. 创建项目

切换到你的工作目录，执行

```
wxp new my_project
```

按照提示完成项目创建。此时在当前目录下就会有一个名为 `my_project` 的目录，里面就是我们的项目源码。

6. 运行创建的项目

在命令行中执行

```
cd my_project  
cnpm i  
npm run serve
```

然后打开浏览器前往 `http://localhost:7777` 就可以看到项目运行的效果，使用playground扫描页面上的二维码可以在手机中查看运行效果。

7. 调试

还是在命令行中执行

```
npm run debug
```

此时会打开调试设备列表页面，之后的操作和官网一样，具体调试方法请参考[官网说明](#)。

net

网络访问

API

```
/** post请求
 * @param url 地址
 * @param param 参数
 * @param header header
 * @param start 请求开始的回调
 * @param success status==200的回调
 * @param complete 请求完成的回调, 不论成功失败
 * @param exception status!=200的回调
 */
post(url,param, header,start, success, complete, exception)

/** get请求
 * @param url 地址
 * @param param 参数
 * @param header header
 * @param start 请求开始的回调
 * @param success status==200的回调
 * @param complete 请求完成的回调, 不论成功失败
 * @param exception status!=200的回调
 */
get(url,param,header,start,success,complete,exception)

/** 
 *
 * @param url 地址
 * @param param 参数
 * @param header header
 * @param path 文件地址可以有多个
 * @param start 请求开始的回调
 * @param success status==200的回调
 * @param complete 请求完成的回调, 不论成功失败
 * @param exception status!=200的回调
 */
postFile(url,param,header,path,start,success,complete,exception)
```

Demo

```
post()
{
```

```
var self=this;
self.back="";
const net = weex.requireModule('net');
net.post('http://121.40.81.1:9080/edu/getBanners.do',{a:"1",b:"2"},{},function(){
    //start
},function(e){
    //success
    self.back=e.res;
    self.header=r.headers;
},function(e){
    //exception

},function(){
    //compelete
});
}

},
get()
{

var self=this;
const net = weex.requireModule('net');
self.back="";
net.get('http://121.40.81.1:9080/edu/getBanners.do',{},{},function(){
    //start
},function(e){
    //success
    self.back=e.res;
},function(e){
    //exception

},function(){
    //compelete
});
}

}
```

navigator

众所周知，在浏览器里，我们可以通过前进或者回退按钮来切换页面，iOS/Android 的navigator模块就是用来实现类似的效果的。除了前进、回退功能，该模块还允许我们指定在切换页面的时候是否应用动画效果。

API

导航跳转

注意：使用push, pushParam, pushFull打开的页面只能用back, backFull返回，使用present打开的页面，只能用dismiss关闭

push打头的方法和present方法打头的方法区别是动画效果不一样，push是由右往左打开一个页面，present是由下往上打开

```
/**  
 * @param url 相对地址  
 */  
push(url)  
  
/**  
 * @param url 相对地址  
 * @param param 参数  
 */  
pushParam(url,param )  
  
/**  
 * @param url 相对地址  
 * @param param 参数  
 * @param navbarVisibility 原生导航栏是否可见（已废弃）  
 * @param callback 页面返回时的回调  
 * @param animate 使用动画（android无效）  
 */  
pushFull(url,param,navbarVisibility,callback,animate)  
  
/**  
 * 关闭当前页面，返回上一个页面，  
 */  
back()  
  
/**  
 * 关闭当前页面，返回上一个页面，同时带上参数  
 * @param param 带给上一个页面的参数  
 * @param animate 是否使用动画，（android无效）  
 */  
backFull(param,animate)
```

```
/**  
 * 打开一个页面，动画是从下往上弹出来  
 */  
present()  
  
/**  
 * 打开一个页面，动画是从下往上弹出来  
 * @param url 相对地址  
 * @param param 参数  
 * @param navbarVisibility 原生导航栏是否可见（后面会废弃）  
 * @param createnav 是否创建导航控制器，默认是（后面会废弃）  
 * @param callback 页面返回时的回调  
 * @param animate 使用动画（android无效）  
 */  
presentFull(url,param, navbarVisibility, createnav,callback,animate)  
  
/**  
 * 关闭， present开头方法打开的页面  
 */  
dismiss()  
  
/**  
 * 关闭， present开头方法打开的页面，同时带上参数  
 * @param param 带给上一个页面的参数  
 * @param animate 是否使用动画，（android无效）  
 */  
dismissFull(param,animate)
```

获取参数

注意：此方法只有写在onPageInit里面才能有效获取

```
/**  
 * 获取上一个页面传递来的参数，  
 */  
param()
```

定页返回

如果你打开的了多个页面，现在希望越过中间的页面，返回第一个或其它页面，这里的方法将帮你做到

```
/**  
 * 返回指定id的页面， id由setPageId方法设定  
 */  
backTo(id)
```

```
/**
设置当前页的id
*/
setPageId(id)

/**
将此页面及这个页面之后打开的页面纳入一个堆栈,
*/
setRoot(id)

现在来说一下怎么达到效果，假设有A->B->C三个页面，现在希望从C直接回到A
1. 在A的onPageInit中调用setRoot('A'), setPageId('A')
2. 在C中backTo('A')
```

Demo

```
/**推入式跳转
push()
{
    var nav = weex.requireModule('navigator') ;
    //相对路径写法，也可以root:绝对路径
    nav.push('page1.js')
}

/**带参数推入式跳转
pushWidthParam()
{
    var self=this;
    var nav = weex.requireModule('navigator') ;
    var param={};
    param.a='value1'
    nav.pushParam('page1.js',param);
}

/**弹出式跳转
present(){
    var self=this;
    var nav = weex.requireModule('navigator') ;
    nav.present('page1.js')
},

/**获取参数
created:function(){

    var self=this;
    var globalEvent = weex.requireModule('globalEvent') ;
    globalEvent.addEventListener("onPageInit", function (e) {
```

```
var navigator = weex.requireModule('navigator') ;
var param=navigator.param();

});

}
```

notify

基于原生的通知组件，你不需要持有目标组件的引用，即可通信，android基于eventbus实现，ios基于系统的notification实现

API

```
/**注册
*/
regist(key,callback)

/**
发送
*/
send(key,param)

/**
发送给原生
*/
sendNative(key,param)
```

Demo

```
send()
{
    var p={};
    p.v1='v1'
    var notify=weex.requireModule("notify")
    notify.send('key',p)
}

//写在其它页面created即可
created:function(){

    const notify = weex.requireModule('notify');
    notify.regist("key",function (res) {
        var v= res.v1;
    });

}
```


photo

用于获取拍照和相册读取照片

API

默认的选择器，包含了相册和相机入口

```
/**  
 * @param width 照片宽度  
 * @param height 照片高度  
 * @param themeColor 相册选择页的背景色  
 * @param titleColor 相册选择页的标题文字颜色  
 * @param cancelColor 相册选择页的取消文字颜色  
 * @param callback 回调，返回图片地址  
 */  
open( width, height , themeColor, titleColor, cancelColor, callback)
```

单独打开相册

```
/**  
 * @param width 照片宽度  
 * @param height 照片高度  
 * @param themeColor 相册选择页的背景色  
 * @param titleColor 相册选择页的标题文字颜色  
 * @param cancelColor 相册选择页的取消文字颜色  
 * @param callback 回调，返回图片地址  
 */  
openPhoto( width, height, themeColor, titleColor, cancelColor, callback )
```

单独打开相机

```
/**  
 *  
 * @param width 照片宽度  
 * @param height 照片高度  
 * @param themeColor 选择页的主题色  
 * @param callback 回调，返回图片地址  
 */  
openCamera( width, height, themeColor, callback )
```

Demo

一般这个组件都是配合上传一起使用

```
//打开全部  
openAll()  
{
```

```

var self=this;
const photo = weex.requireModule('photo');
photo.open(500,800,'#000000','#ffffff','#ffffff',function(e){
self.src=e.path;
var param={};
var header={};
var path={};
path.file=e.path;
var net=weex.requireModule("net");
net.postFile('http://xxx/upload',param,header,path,()=>{
//start
},(e)=>{
//success
var modal=weex.requireModule("modal")
modal.toast({message:'上传成功! '})
},()=>{
//complete

},()=>{
//exception
var modal=weex.requireModule("modal")
modal.toast({message:'上传异常! '})
})
});

}

//打开相册
openPhoto()
{
var self=this;
const photo = weex.requireModule('photo');
photo.openPhoto(500,800,'#000000','#ffffff','#ffffff',function(e){

self.src=e.path;
var param={};
var header={};
var path={};
path.file=e.path;
var net=weex.requireModule("net");
net.postFile('http://xxx/upload',param,header,path,()=>{
//start
},(e)=>{
//success
var modal=weex.requireModule("modal")
modal.toast({message:'上传成功! '})
},()=>{
//complete

```

```
},()=>{
    //exception
    var modal=weex.requireModule("modal")
    modal.toast({message:'上传异常! '})
}

});

},

//打开相机
openCamera()
{
    var self=this;
    const photo = weex.requireModule('photo');
    photo.openCamera(500,800,'#000000',function(e){

        self.src=e.path;
        var param={};
        var header={};
        var path={};
        path.file=e.path;
        var net=weex.requireModule("net");
        net.postFile('http://xxx/upload',param,header,path,()=>{
            //start
            },(e)=>{
            //success
            var modal=weex.requireModule("modal")
            modal.toast({message:'上传成功! '})
            },()=>{
            //complete

            },()=>{
            //exception
            var modal=weex.requireModule("modal")
            modal.toast({message:'上传异常! '})
            })
    });

},
```

static

用于存储全局变量，只要app不被杀死就会存在，用于存储登录后的user最合适了

API

```
/**
 * 存储对象
 * @param key
 * @param value
 */
set(key,value)

/**
 * 获取对象
 * @param key
 */
get(String key)

/**
 * 存储字符串
 * @param key
 * @param value
 */
setString(key,value)

/**
 * 获取字符串
 * @param key
 */
getString(String key)
```

Demo

```
saveString()
{
    var pref=weex.requireModule("static")
    pref.setString('key',1111);
    var modal=weex.requireModule("modal")
    modal.toast({message:'存储成功'});
}

getString()
{
    var pref=weex.requireModule("static")
    var s= pref.getString('key');
```

```
var modal=weex.requireModule("modal")
modal.toast({message:'存储成功的值: '+s});
},

saveObj()
{
    var pref=weex.requireModule("static")
    var obj={};
    obj.a=1;
    obj.b=2;
    obj.c=3;
    pref.set('objkey',obj);
    var modal=weex.requireModule("modal")
    modal.toast({message:'存储成功'});
    this.getObj();
},
getObj()
{
    var pref=weex.requireModule("static")
    var p= pref.get('objkey');
    this.data=p;
},
```

pref

持久存储，即使app关闭之后也会存在，类似于cookie, android基于sharerefrence, Ios基于NSUserDefaults

API

```
/**
 * 存储对象
 * @param key
 * @param value
 */
set(key,value)

/**
 * 获取对象
 * @param key
 */
get(String key)

/**
 * 存储字符串
 * @param key
 * @param value
 */
setString(key,value)

/**
 * 获取字符串
 * @param key
 */
getString(String key)
```

Demo

```
saveString()
{
    var pref=weex.requireModule("pref")
    pref.set('key',this.text);
    var modal=weex.requireModule("modal")
    modal.toast({message:'存储成功'});
}

getString()
{
```

```
var pref=weex.requireModule("pref")
var s= pref.get('key');
var modal=weex.requireModule("modal")
modal.toast({message:'存储成功的值'+s});
},
remove()
{
    var pref=weex.requireModule("pref")
    pref.remove('key')
    pref.remove('objkey')
    var s= pref.get('key');
    var modal=weex.requireModule("modal")
    modal.toast({message:'删除成功: '+s});
},
saveObj()
{
    var pref=weex.requireModule("pref")
    var obj={};
    obj.a=1;
    obj.b=2;
    pref.setObj('objkey',obj);
    var modal=weex.requireModule("modal")
    modal.toast({message:'存储成功'});
},
getObj()
{
    var pref=weex.requireModule("pref")
    var p= pref.getObj('objkey');
    this.data=p;
//        var modal=weex.requireModule("modal")
//        modal.toast({message:p});
},
}
```

fpicker

picker组件，最多三列，可简单定制外观

API

注意：setCount必须第一个调用，以初始化picker

设置picker有几列

```
/**  
 * @param count 列数  
 */  
setCount(count)
```

设置第一列的数据

```
/**  
 * @param list 数据  
 */  
setItems1(list)
```

设置第二列的数据

```
/**  
 * @param list 数据  
 */  
setItems2(list)
```

设置第三列的数据

```
/**  
 * @param list 数据  
 */  
setItems3(list)
```

设置主题色

```
/**  
 * @param bgcolor 背景色  
 * @param btncolor 按钮文字颜色  
 */  
setTheme( bgcolor, btncolor)
```

```
/*  
 * 显示picker  
 */
```

```
show()

用户选中项目时调用此方法设置的回调

/**
 * 回调
 * @param callback
 */
setDone(callback)
```

三个滚轮滚动时触发的回调，可在此设置关联关系

```
/**
 *
 * @param change1
 * @param change2
 * @param change3
 */
setChange( change1, change2, change3)

/**
 * 选中某一列某一行
 * @param c 列数
 * @param row 行数
 *
 */
select( c, row)
```

Demo

```
const data=require('./data.js')

export default {

  get(res)
  {
    var self=this;
    var picker=weex.requireModule("fpicker");
    var d=data.getAddressData();
    var l= d.list
    //务必调用此方法，此方法有初始化的功能，最多3列
    picker.setCount(3);
    picker.setItems1(this.toArray(l))
    picker.setItems2(this.toArray(l[0].children))
    picker.setTheme('#f9f9f9','#4c4c4c')

    var index1=0;
    var index2=0;
```

```

var index3=0;
//每个滚轮的change事件
picker.setChange(function(e){
    index1=e.index;
    picker.setItems2(self.toArray(l[index1].children))
    picker.select(2,0);
    if(l[index1].children.length>0)
    {
        picker.setItems3(self.toArray(l[index1].children[0].children))

        picker.select(3,0);
    }
    else
    {
        picker.setItems3([])
    }

},function(e){
    index2=e.index;
    if(l[index1].children[index2].children.length>0)
    {
        picker.setItems3(self.toArray(l[index1].children[index2].children))
        picker.select(3,0);
    }
    else
    {
        picker.setItems3([])
    }
},function(e){
    index3=e.index;
})
//点击完成的事件
picker.setDone(function(e){
    var p1= l[e.index1];
    var p2= l[e.index1].children[e.index2];
    var p3= l[e.index1].children[e.index2].children[e.index3];
    res(p1,p2,p3)

});
// picker.show()
return picker;
},
toArray(list)
{
    var p=[];
    for(let i=0;i<list.length;i++)
    {
        p.push(list[i].name);
    }
}

```

```
    }
    return p;
},
}

}
```

progress

一个简单的无状态进度指示器

API

```
/**
 * 显示
 */
show()

/**
 * @param txt 加载的文字
 */
showFull(txt)

/**
 *关闭
 */
dismiss()
```

Demo

```
post()
{
    var self=this;
    self.back="";
    const net = weex.requireModule('net');
    const progress = weex.requireModule('progress');
    net.post('http://121.40.81.1:9080/edu/getBanners.do',{a:"1",b:"2"},{},function(){
        //start
        progress.show()
    },function(e){
        //success
    },function(e){
        //exception

    },function(){
        //complete
        progress.dismiss()
    });
},
```


qr

二维码扫描组件

API

```
/**  
 * 存储对象  
 * @param param 设置二维码扫描界面的参数  
 * @param callback 回调函数  
 */  
open( param,  callback )
```

Demo

```
open()  
{  
  var qr=weex.requireModule('qr')  
  var p={};  
  p.color='#000000'  
  p.bgcolor='#ffffff'  
  qr.open(p,(res)=>{  
    var url=res.url  
  })  
}
```

addressBook

获取通讯录联系人信息

API

```
/**  
 *读取联系人信息  
 */  
read(callback)
```

Demo

```
read()  
{  
    var book=weex.requireModule('addressBook')  
  
    book.read((res)=>{  
        var people=res.people;  
        var p0=people[0];  
        var firstname=p0.firstName  
        var lastName=p0.lastName  
        var phones=p0.phones  
  
    })  
}
```

net

网络请求

API

```
/**  
 * @param url 地址  
 * @param param 参数  
 * @param header header  
 * @param start 开始的回调  
 * @param success status==200的回调  
 * @param complete 完成的回调, 不论请求成功失败  
 * @param exception status!=200的回调  
 */  
post(url, param, header, start, success, complete, exception)  
  
/**  
 * @param url 地址  
 * @param param 参数  
 * @param header header  
 * @param start 开始的回调  
 * @param success status==200的回调  
 * @param complete 完成的回调, 不论请求成功失败  
 * @param exception status!=200的回调  
 */  
get(url, param, header, start, success, complete, exception)  
  
/**  
 * @param url 地址  
 * @param param 参数  
 * @param header header  
 * @param path path  
 * @param start 开始的回调  
 * @param success status==200的回调  
 * @param complete 完成的回调, 不论请求成功失败  
 * @param exception status!=200的回调  
 */  
postFile(url, param, header, path, start, success, complete, exception)
```

Demo

```

post()
{
    var self=this;
    self.back="";
    const net = weex.requireModule('net');
    net.post('http://121.40.81.1:9080/edu/getBanners.do',{a:"1",b:"2"},{},function(){
        //start
    },function(e){
        //success
        self.back=e.res;
        self.header=r.headers;
    },function(e){
        //exception

    },function(){
        //compelete
    });
}

,
get()
{

    var self=this;
    const net = weex.requireModule('net');
    self.back="";
    net.get('http://121.40.81.1:9080/edu/getBanners.do',{},{},function(){
        //start
    },function(e){
        //success
        self.back=e.res;
    },function(e){
        //exception

    },function(){
        //compelete
    });
},
//上传文件
openPhoto()
{
    var self=this;
    const photo = weex.requireModule('photo');
    photo.openPhoto(500,800,'#000000','#ffffff','#ffffff',function(e){

        self.src=e.path;
        var param={};
        var header={};

```

```
var path={};
path.file=e.path;
var net=weex.requireModule("net");
net.postFile('http://xxx/upload',param,header,path,()=>{
    //start
},(e)=>{
    //success
    var modal=weex.requireModule("modal")
    modal.toast({message:'上传成功! '})
},()=>{
    //complete
},()=>{
    //exception
    var modal=weex.requireModule("modal")
    modal.toast({message:'上传异常! '})
})
});
```


looper

文字轮播器

属性	作用
font-size	字体大小
data	数据源(字符串数组)
color	文字颜色
interval	轮播的时间间隔, 单位秒
text-align	文字居中 (center) , 居左(left), 居右(right)

Demo

```
<template>

    <div>
        <div title="相机" style="height:100px" append="tree">
            </div>
        <looper ref="looper" font-size="25" @click="ok" :data="items" color="#eeeeee"
            style="width: 300;height: 100;background-color: #0088fb">
            </looper>
            <text>{{index}}</text>

            <div style="width: 200;height: 100;background-color: #006ce7" @click="getIndex
"></div>
        </div>

    </template>

    <style>
        .cl
        {
            align-items: center;
        }
    </style>
```

```
</style>

<script>

var head =require('./header.vue')

export default {
  components:{head},
  data () {
    return {
      src:"",
      index:0,
      items:['2017年11月9日, 网络设计平台app','2017年11月9日, 网络设计平台app','2017
年11月9日, 网络设计平台app','2017年11月9日, 网络设计平台app','2017年11月9日, 网络设计平台app']
    }
  },
  methods: {
    ok()
    {
      this.$refs.looper.getIndex((res)=>{
        this.index=res.index;
      });

    },
    getIndex()
    {
      this.$refs.looper.getIndex((res)=>{
        this.index=res.index;
      });
    },
    onChange(res)
    {
      this.index=res.index;
    },
  }

  var globalEvent = weex.requireModule('globalEvent') ;

  globalEvent.addEventListener("onPageInit", function (e) {
    const nav = weex.requireModule('navbar');
    nav.setTitle('照相');
    nav.setBack(true);
    nav.setRightImage('img/scan.png',function(res){




```

```
        var modal = weex.requireModule('modal') ;
        modal.alert({message:"ok"})
    });
});

}
</script>
```

host

一个多页面隐藏切换的组件，比如tab组件

属性	作用
items	子页面js地址
index	当前显示那一页

Demo

```

<template>
    <div style="flex: 1;">

        <host :index="index" :items="items" style="position: absolute;left: 0;top: 0;right: 0;bottom: 100;">

        </host>
        <div style="height: 100;width: 750;position: absolute;bottom: 0;left: 0;right: 0;flex-direction: row;background-color: #0088fb">
            <div @click="change(0)" style="flex: 1;align-items: center;justify-content: center">
                <text>推荐</text>
            </div>
            <div @click="change(1)" style="flex: 1;align-items: center;justify-content: center">
                <text>电视剧</text>
            </div>
            <div @click="change(2)" style="flex: 1;align-items: center;justify-content: center">
                <text>电影</text>
            </div>
            <div @click="change(3)" style="flex: 1;align-items: center;justify-content: center">
                <text>收藏</text>
            </div>
        </div>

        <!--<prerender src="app/busi/tab/serial.js"></prerender>-->
        <!--<a href="demo/pagedemo.js" style="width: 100;height: 100;background-color: red;justify-content: center;align-items: center"><text>跳转</text></a>-->
    </div>
</template>

<style>
    .title { padding-top:40px; padding-bottom: 40px; font-size: 48px; }

```

```

.logo { width: 360px; height: 156px; }
.desc { padding-top: 20px; color:#888; font-size: 24px; }
</style>

<script>

export default {
  components:{ },
  data: {
    logoUrl: 'http://img1.vued.vanthink.cn/vued08aa73a9ab65dcbd360ec54659ada97c.png',
    target: 'World',
    index:0,
    items:[ '../../busi/tab/mainpage.js','../../busi/tab/serial.js','../../busi/tab/movie.js','../../demo/hmain.js']
  },
  methods: {
    update: function (e) {
      this.target = 'Weex'
      console.log('target:', this.target)
    },
    change(i)
    {
      this.index=i;
    },
    show()
    {
      var modal=weex.requireModule("modal")
      var p=weex.config.env.osVersion
      //      p=p.replace('.','');
      //      p=p.replace(/./g,"");
      //      p= p.replace(/\./g,'')
      modal.alert({message:p})
    }
  }
}

created:function(){

  var globalEvent = weex.requireModule('globalEvent') ;
  var self=this;
  globalEvent.addEventListener("onPageInit", function (e) {

    const nav = weex.requireModule('navbar');
    nav.hide();

  }));
}

```

```
    },
}
</script>
```

web

浏览器组件

属性	作用
src	链接地址（扩展：可以直接显示字符串）

Demo

```
<template>
<div>

    <div style="flex:1">
        <web :src="src" bounce="false" @pagestart="pagestart" @pagefinish="pagefinish"
@error="pagefinish" style="position: absolute;left: 0;top: 0;right: 0;bottom: 0;background
-color: red">

        </web>
    </div>

</div>

</template>
<style>
.text {
    font-size: 50;
}

.btn{
    background-color:#0085ee;
    height:100;

    margin-top:50;
    margin-left: 50;
    margin-right: 50;
    border-radius:10;
    align-items:center;
    justify-content:center;

}
.btn:active{background-color:#006ce7;}
</style>

<script>
```

```
var progress=weex.requireModule("progress")
export default {
  components:{},
  data () {
    return {
      text: '',
      param:'',
      data:{},
      src:''
    }
  }
,
  methods:{
    load()
    {
      this.src='https://www.baidu.com'
      this.src='<div>这是富文本
```

floating

小菊花进度指示器

属性	作用
color	颜色

Demo

```
<template>
  <div>
    <floating class="indicator" color="#000000" > </floating>
  </div>
</template>
<style>
  .indicator {
    color: #888888;
    height: 40;
    width: 40;
    margin-right: 10;
  }
</style>

<script>

  export default {
    components:{ },
    data () {
      return {

      }
    }
    ,
    methods:{

    }
    ,
    created:function(){

    }
  }
</script>
```


image

图片控件

属性	作用
src	支持root:写法
placeholder	支持root:写法

drawerlayout

抽屉组件

属性	作用
src	主页面地址
slidSrc	侧滑页面地址

Demo

```
<template>
<drawerlayout
ref="drawer"
slid_src="userNav.js"
src="farmWork.js"
>

</drawerlayout>
</template>

<style>
.header {
width: 750px;
}
/*back按钮*/
.siftL {
position: absolute;
left: 0;
top: 40px;
width: 80px;
height: 80px;
z-index: 1000;
}

.icon-navR {
width: 80px;
height: 80px;
z-index: 1000;
}

.

.active {
background-color: blue;
}

.item {
```

```
margin-top: 1;
height: 180;
align-items: center;
background-color: #ffffff;
}

.item:active {
background-color: #e2e2e2;
}

.farmImage {
width: 60px;
height: 60px;
margin-bottom: 10px;
}

.text {
font-size: 28px;
color: #7f7f7f;
}
</style>
<script>
var head = require('../component/header.vue');
var flist = require('../component/flist.vue');
export default {
components: {head, flist},
data(){
return {
items: [
{"id":0,"url": "../img/crop1.png", "name": "玉米"},
 {"id":1,"url": "../img/crop2.png", "name": "马铃薯"},
 {"id":2,"url": "../img/crop3.png", "name": "大豆"},
 {"id":3,"url": "../img/crop4.png", "name": "水稻"},
 {"id":4,"url": "../img/crop5.png", "name": "小麦"},
 {"id":5,"url": "../img/crop6.png", "name": "花生"},
 {"id":6,"url": "../img/crop7.png", "name": "大豆"},
 {"id":7,"url": "../img/crop8.png", "name": "谷子"},
 {"id":8,"url": "../img/crop9.png", "name": "牧草"},
],
}
},
methods: {

crop_detail(item){
var nav=weex.requireModule('navigator')
nav.pushParam('cropList.js',item);

},
refresh: function () {
```

```
},
},
,
created: function () {

var notify=weex.requireModule("notify")
notify.regist('toggle',()=>{
this.$refs.drawer.toggle()
})

var globalEvent = weex.requireModule('globalEvent');
globalEvent.addEventListener("onPageInit", function(e) {

notify.sendNative('entry',{})

});
}
}
</script>
```

root:

当写一个vue组件的时候，最终计算相对路径是按组件宿主的位置来计算的，这时候就会计算错误，所以我们新增root:来解决此问题

哪些地方支持：

image:src,placeholder属性

navigator: 所有url参数

embed:src属性