API文档

1. 配置(AppConfig)

SERVER\_URL:’’ //服务器地址，与请求服务中的url参数拼接为完整url

DEFAULT\_PARAMS:{ //每个请求都会传递的公共参数

APP\_SERSION:’0.0.3’,

…

},

1. 服务
2. 缓存服务(AppCacheService)

|  |  |  |
| --- | --- | --- |
| 方法名 | 方法描述 | 参数 |
| setMemoryCache(key, value) | 将数据存入内存缓存（退出应用后即消失） | key：保存到内存的键值，用于取值；  value：存到内存中的值 |
| getMemoryCache(key) | 根据键值从内存中取出存入的值 | Key：保存到内存的键值 |
| removeMemoryCache(key) | 根据键值清除内存缓存 | Key：保存到内存的键值 |
| setStorageCache(key, value) | 将数据存入本地缓存（清除缓存之前一直存在） | key：保存到缓存的键值，用于取值；  value：存到内存中的值 |
| getStorageCache(key) | 根据键值从缓存中取出存入的值 | key：保存到缓存的键值 |
| removeStorageCache(key) | 根据键值清除本地缓存 | key：保存到缓存的键值 |

1. 请求服务(AppHttpService)

|  |  |  |
| --- | --- | --- |
| 方法名 | 方法描述 | 参数 |
| Send(config) | 向后台发送请求 | {  url : ‘’, //请求的链接,  showLoading:true, //是否显示遮罩（默认显示）  timeout:3000, //显示遮罩最大时间（默认30秒）  params:{ //请求的参数  param1:’’,  param2:{}  },  onBefore:function(para){},//发送请求前处理方法(参数为发送请求处理后的参数)  onSuccess : function(result){  //请求成功时回调  },  onError:function(){  //请求异常时回调，如网络异常、后台返回的格式不正确导致解析异常  }  } |

1. 消息服务（AppMessageService）

|  |  |  |
| --- | --- | --- |
| 方法名 | 方法描述 | 参数 |
| isPopupShown() | 判断当前是否有对话框 | 无 |
| closePopup() | 关闭当前显示的对话框 | 无 |
| appToast(config) | 显示通知，形式为toast悬浮框 | {  content:’’,//显示消息内容  delay:0,//延迟几秒后显示（默认为0）,  duration:3000,//显示几秒后消失（默认为3秒）  } |
| alert(config) | 提示框 | {  title:’消息’,//对话框标题（默认为“消息”）  content:’’,//对话框内容（可以为html模板）  okText:’确定’,//确定按钮内容（默认为“确定”）  onSelect:function(){},//点击确认后回调  } |
| confirm(config) | 确认框 | {  title:’消息’,//对话框标题（默认为“消息”）  content:’’,//对话框内容（可以为html模板）  okText:’确定’,//确定按钮内容（默认为“确定”）  cancelText:’取消’,//取消按钮内容（默认为“取消”）,  onSelect:function(res){}//点击按钮后回调，true为确认，false为取消  } |
| dialog(config) | 自定义对话框 | {  title:’消息’,//对话框标题（默认为“消息”）  content:’’,//对话框内容（可以为html模板）  buttons:[{  text:’按钮1’,//按钮文字  class:’’,//按钮样式（默认为确认样式）  onSelect:function(event){}//选择按钮后回调，回调参数为事件  },{  text:’按钮2’,  class:’’,  onSelect:function(event){}  }]  } |

1. 监听服务（AppListenerService）

监听事件：

1. 物理返回键（LISTENER\_TYPES.DEVICE.BACK\_BUTTON）

回调参数：

{

stopAction:function(){} //停止默认物理返回键行为

}

|  |  |  |
| --- | --- | --- |
| 方法名 | 方法描述 | 参数 |
| regist(config) | 注册监听事件 | {  page:’home\_tab’,//页面路由,  when:’’,//监听事件  action:function(param){}//触发事件后回调方法  } |

1. 工具服务（AppUtilsService）

|  |  |  |
| --- | --- | --- |
| 方法名 | 方法描述 | 参数 |
| interval(config) | 定时器，返回值为定时器对象，可用于停止定时器 | {  action:function(){},//定时器执行的函数  time:1000//定时器循环周期  } |
| cancelInterval(timer); | 停止定时器 | 定时器对象 |
| checkPhoneNum(phoneNum) | 检查电话号码格式，返回值：  -1：格式错误  0：传入电话号码为空，  1：电话号码格式正确 | 电话号码 |
| updateVersion(callback) | 检查更新 | 检查更新后，无更新时的回调方法 |

1. 指令
2. 列表选择器（AppListPickerDirective）

使用方法：

<div app-list-picker-directive title={{pickerTitle}} data="pickerItems"  
 select-finish="selectItem(params)" bind="bind(params)"></div>

controller中使用方法：

//定义选择器标题

$scope. pickerTitle = ‘选择器标题’;

$scope. pickerItems = [{

text:’选项1’,//必填，作为展示用

value:’选项值1’,//必填，作为选项唯一标识

testValueObject1:{},//键值自定义

testValueObject2:[],

…

}]

//绑定

$scope.bind = function(param){

/\* 该showPicker为自己定义的方法，绑定后，调用定义的方法即可调出选择器组件\*/

showPicker = param.show;

}

//选择回调

$scope. selectItem = function(param){

//回调参数用法

var paramText = param.item.text;

var paramValue = param.item.value;

var paramObject = param.item.testValueObject1;

}

|  |  |  |
| --- | --- | --- |
| 方法名 | 方法描述 | 参数 |
| bind(param) | 绑定数据或方法 | param:{  show:function(){}//显示组件  } |
| selectItem(param) | 选择后回调方法 | param为选项内容 |