# Angular04

## 复习

* 服务
* 相当于 vue 中的 Vuex. 全局的状态管理. 组件间的数据共享.
* 生成服务: ng g s 服务名
* 使用服务: 依赖注入机制
* class Demo{  
   //声明依赖: Demo类的实例化操作 必须传递一个 string 类型的值  
   constructor(name: string, public xxx: 服务类){  
   // 服务的语法糖: 权限词 变量名: 类型名  
   // 用于快速引入服务  
   }  
  }  
    
  // 使用时, 必须传递依赖的  
  new Demo('mike')
* 网络服务
* 网络服务由 angular 提供的 网络模块.
* 网络模块 和 forms 模块相同, 都必须手动引入注册模块 才能使用
* 使用方式: this.http.get(url).subscribe( res=> {} )
* 网络服务的跨域, 前端通过 proxy 代理解决.
* 

## 父子传参

在 vue 中:

* 在 vue 中, 子组件 通过 props : ['name'] 声明属性
* 使用时: <tag name='东东' />

生成 myc01 组件

<!-- 父子传参 -->  
<!-- js才有数据类型, html 都是字符串 -->  
<!-- [属性名]="js代码" -->  
<app-myc01  
 name="东东"  
 [age]="33"  
 [boss]="{ name: 'wenhua', age: 44 }"  
></app-myc01>

import { Component, Input, OnInit } from '@angular/core';  
  
@Component({  
 selector: 'app-myc01',  
 templateUrl: './myc01.component.html',  
 styleUrls: ['./myc01.component.css'],  
})  
export class Myc01Component implements OnInit {  
 // 声明 要接受 name 属性  
 // vue中: props: ['name']  
 // @Input(): 固定标识, 代表 name 属性是外来的  
 @Input() name: string;  
 @Input() age: number;  
  
 // object 相当于 {} 空对象  
 // 对象类型 要自己声明  
 @Input() boss: Boss;  
  
 constructor() {  
 this.boss.name;  
 }  
  
 ngOnInit(): void {}  
}  
  
///////////////////////////////////  
//////////////对象类型要 自定义/////  
interface Boss {  
 name: string;  
 age: number;  
}

## 子父传参

在vue中: 父传递一个事件(函数)给子, 子通过 $emit() 触发传入的函数, 函数的参数是子的.. 即 函数是父的, 参数是子的, 实现子的值 传递给父

例如: 身在曹营心在汉-- 关羽就是函数, 内部的this 指向汉. 但是关羽在曹操. 获取 曹操的消息 然后通过内部的 this 传递给 汉.

生成组件 myc02

子父传参4个步骤:

1. 父声明方法
2. 通过事件方式 传递给子
3. 子声明 接收传入的事件
4. 子 触发传入的事件, 传入参数

总结: 方法是父的, 参数是子的

父ts

import { Component } from '@angular/core';  
  
@Component({  
 selector: 'app-root',  
 templateUrl: './app.component.html',  
 styleUrls: ['./app.component.css'],  
})  
export class AppComponent {  
 title = 'ngpro';  
  
 show(msg) {  
 console.log(msg);  
 }  
}

父html

<!-- 子父传参 -->  
<!--   
 1. 父要声明一个函数, 通过事件方式 传递给子  
 2. 子 接收 父传入的 事件函数  
 3. 子 通过 emit 触发函数, 在函数中传入自己的参数  
 -->  
<!-- msgEvent: 需要在子中声明 -->  
<!-- $event: 是固定参数 -->  
<app-myc02 (msgEvent)="show($event)"></app-myc02>

子ts

import { Component, EventEmitter, OnInit, Output } from '@angular/core';  
  
@Component({  
 selector: 'app-myc02',  
 templateUrl: './myc02.component.html',  
 styleUrls: ['./myc02.component.css'],  
})  
export class Myc02Component implements OnInit {  
 // 接收事件的属性  
 // input 向内传递 output 向外传递  
 // EventEmitter: 事件触发器. 传入的show方法 保存在这个对象里  
 // 传入的 show() 就会存放在 msgEvent 对象里  
 @Output() msgEvent = new EventEmitter();  
  
 constructor() {}  
  
 ngOnInit(): void {}  
}

子 html

<p>myc02 works!</p>  
  
<div>  
 <!-- emit(): 触发 msgEvent 中保存的函数, 即 show 函数 -->  
 <button (click)="msgEvent.emit('大桥')">大桥</button>  
 <button (click)="msgEvent.emit('蔡文姬')">蔡文姬</button>  
</div>

## 掌控子元素

vue 中对应的概念: ref 本质就是把 标签 和 变量 绑定在一起.

<tag ref='变量' />

通过 this.refs.变量 就可以使用 绑定的元素

生成组件 myc03

<!-- 掌控子元素 -->
  
<!-- 使用 # 声明唯一标识, 类似id -->
  
<app-myc03 #mm3></app-myc03>
  
   
<button (click)="test()">点击</button>

import { Component, ViewChild } from '@angular/core';
  
import { Myc03Component } from './myc03/myc03.component';
  
   
@Component({
  
 selector: 'app-root',
  
 templateUrl: './app.component.html',
  
 styleUrls: ['./app.component.css'],
  
})
  
export class AppComponent {
  
 // 关联变量 与 子元素
  
 // ViewChild(): 查找到唯一标识对应的元素绑定给后面的变量上
  
 @ViewChild('mm3') suibian: Myc03Component; //补类型, 有代码提示
  
   
 test() {
  
 console.log(this.suibian);
  
 this.suibian.age += 2;
  
   
 this.suibian.show();
  
 }
  
   
 title = 'ngpro';
  
   
 show(msg) {
  
 console.log(msg);
  
 }
  
}

## ionic

官网: https://ionicframework.com/

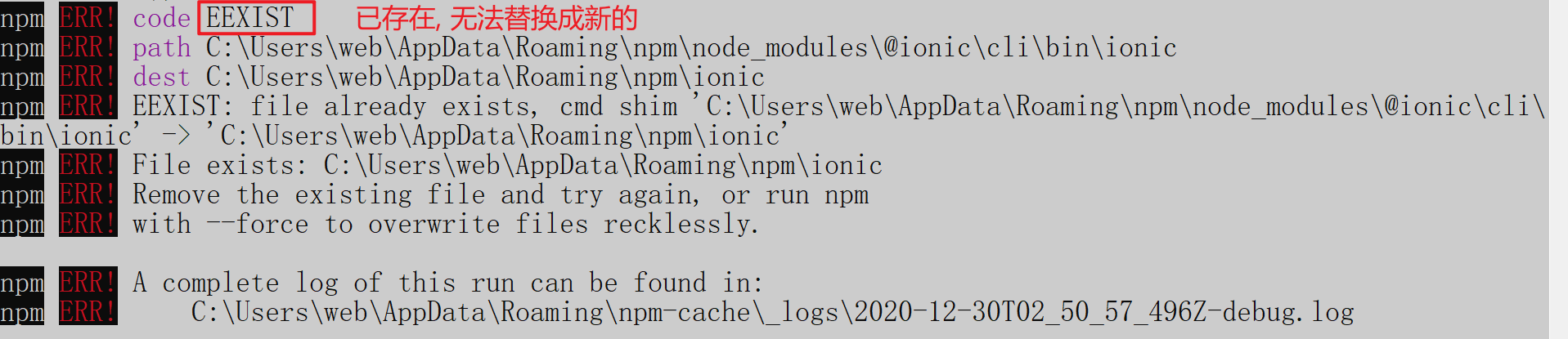
vue 开发手机端库, 有 mintUI 库, vantUI 库, uniapp 库

vue 开发网页端库: elementUI, antd -- ant design 蚂蚁--阿里

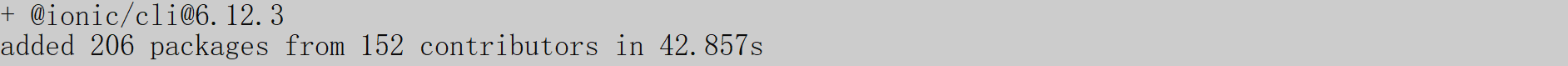
angular 典型的 手机端UI库: **ionic**

安装 ionic 脚手架:

npm i -g @ionic/cli



报错的解决方案: --force表示 强制替换  
npm i -g @ionic/cli --force



生成包命令: **注意cmd开启的目录** 包在此目录下生成

ionic start 包名 应用类型(blank/tabs/sidemenu)  
  
\* blank: 空包, 最基础的  
\* tabs: 带有 标签导航栏, 例如 微信下方的导航栏  
\* sidemenu: 带有侧边栏的  
  
此处创建最基础的  
ionic start ionicApp blank  
  
过程中的选项 都回车 即可

没有生成成功的同学, 3个方式下载

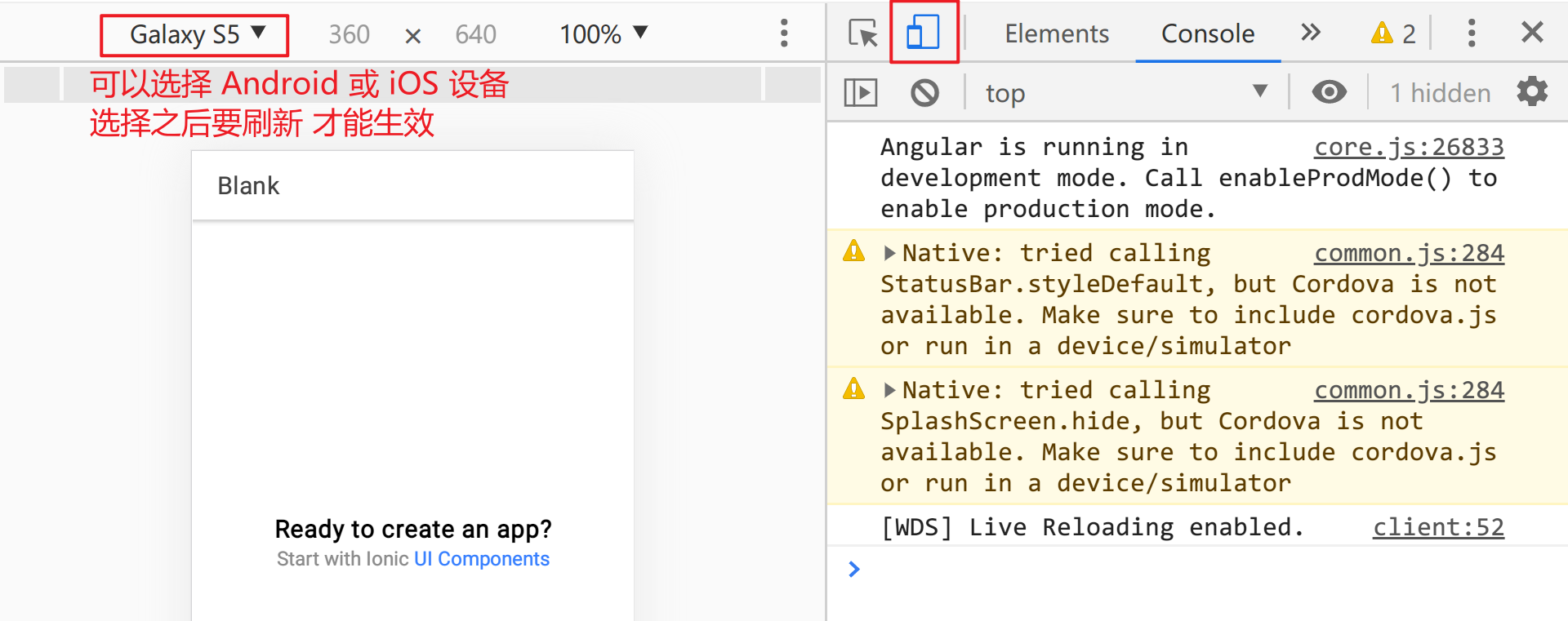
* 微信中
* FTP 的 angular
* 百度网盘
* 链接：https://pan.baidu.com/s/1pLIShrdqlDwXAMC5k24Shw   
  提取码：ugc7

### 组件库

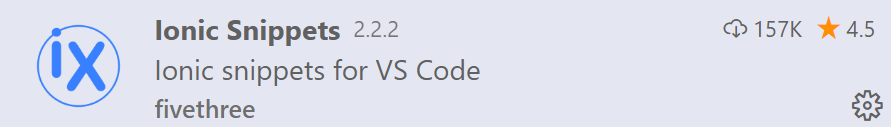
https://ionicframework.com/docs/components

项目的启动命令: 启动的是 **8100** 端口

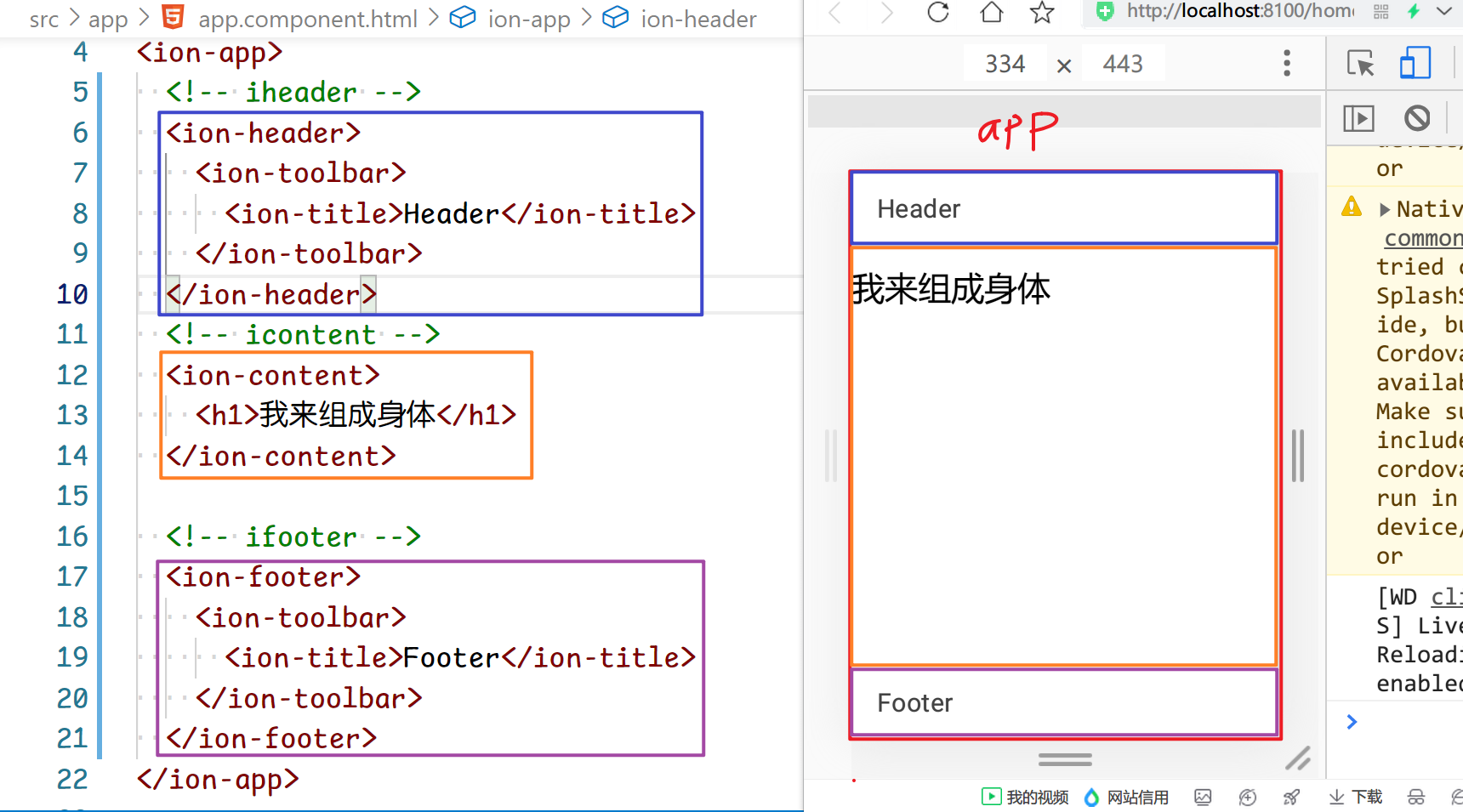
ionic s
  
   
如果脚手架没安装成功的同学:
  
npx ionic s



### 插件



### 容器



### 按钮组件

https://ionicframework.com/docs/api/button

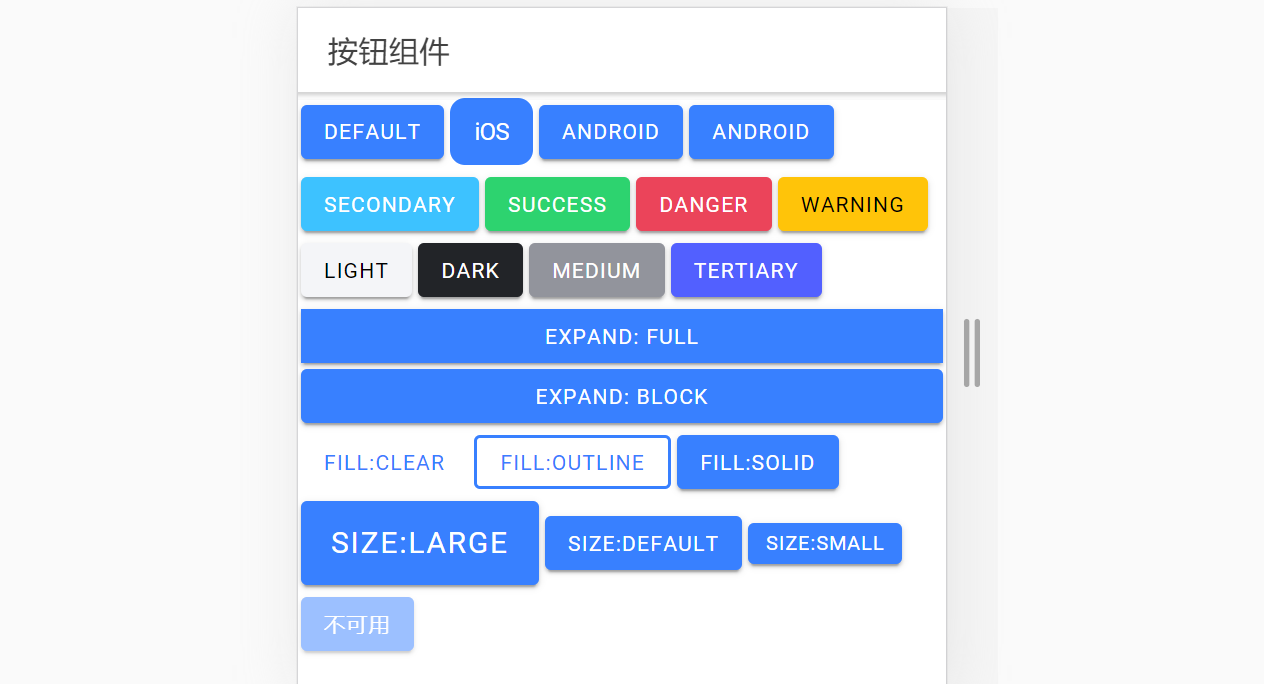
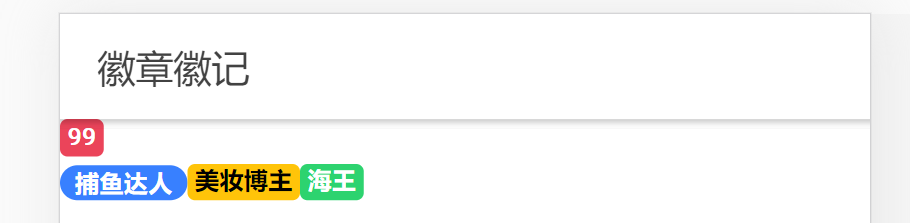


image-20201230150833231

<!-- 按钮组件 -->
  
<ion-app>
  
 <!-- iheader -->
  
 <ion-header>
  
 <ion-toolbar>
  
 <ion-title>按钮组件</ion-title>
  
 </ion-toolbar>
  
 </ion-header>
  
   
 <ion-content>
  
 <!-- 按钮 -->
  
 <ion-button> default </ion-button>
  
   
 <!-- 风格: 默认UI风格 随系统自动适配.
  
 可以通过 mode 属性指定风格 -->
  
 <ion-button mode="ios"> iOS </ion-button>
  
 <ion-button mode="md"> android </ion-button>
  
   
 <!-- 色彩主题: 官方提供了一些主题色 -->
  
 <ion-button color="primary"> android </ion-button>
  
 <ion-button color="secondary"> secondary </ion-button>
  
 <ion-button color="success"> success </ion-button>
  
 <ion-button color="danger"> danger </ion-button>
  
 <ion-button color="warning"> warning </ion-button>
  
 <ion-button color="light"> light </ion-button>
  
 <ion-button color="dark"> dark </ion-button>
  
 <ion-button color="medium"> medium </ion-button>
  
 <ion-button color="tertiary"> tertiary </ion-button>
  
   
 <!-- 按钮的延展 expand -->
  
 <ion-button expand="full"> expand: full </ion-button>
  
 <ion-button expand="block"> expand: block </ion-button>
  
   
 <!-- 按钮的填充 -->
  
 <ion-button fill="clear"> fill:clear </ion-button>
  
 <ion-button fill="outline"> fill:outline </ion-button>
  
 <ion-button fill="solid"> fill:solid </ion-button>
  
   
 <!-- 按钮的大小 -->
  
 <ion-button size="large"> size:large </ion-button>
  
 <ion-button size="default"> size:default </ion-button>
  
 <ion-button size="small"> size:small </ion-button>
  
   
 <!-- 不可用 -->
  
 <ion-button disabled> 不可用 </ion-button>
  
 </ion-content>
  
</ion-app>

### 徽章

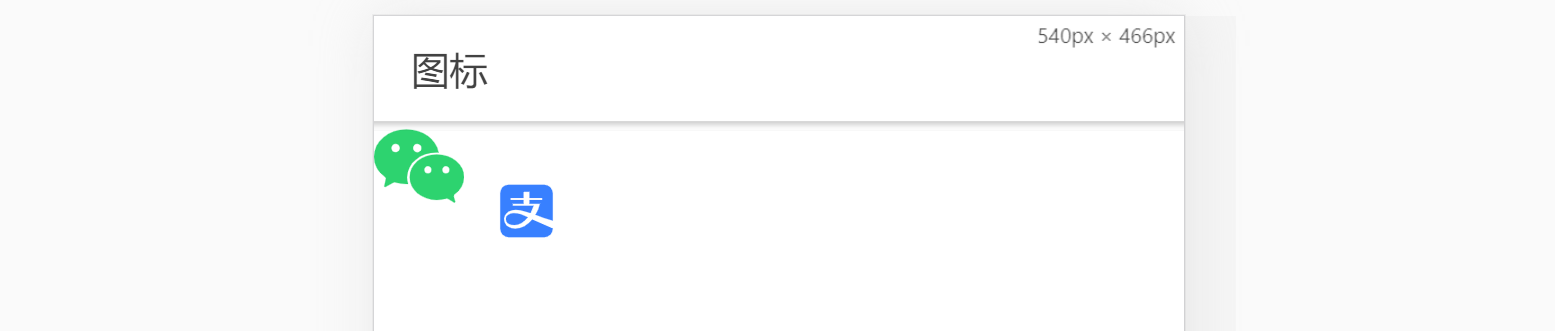
https://ionicframework.com/docs/api/badge



<!-- 徽章徽记 组件 -->
  
<ion-app>
  
 <ion-header>
  
 <ion-toolbar>
  
 <ion-title>徽章徽记</ion-title>
  
 </ion-toolbar>
  
 </ion-header>
  
   
 <ion-content>
  
 <ion-badge color="danger">99</ion-badge>
  
 <br />
  
 <ion-badge color="primary" mode="ios">捕鱼达人</ion-badge>
  
 <ion-badge color="warning">美妆博主</ion-badge>
  
 <ion-badge color="success">海王</ion-badge>
  
 </ion-content>
  
</ion-app>

### 图标组件

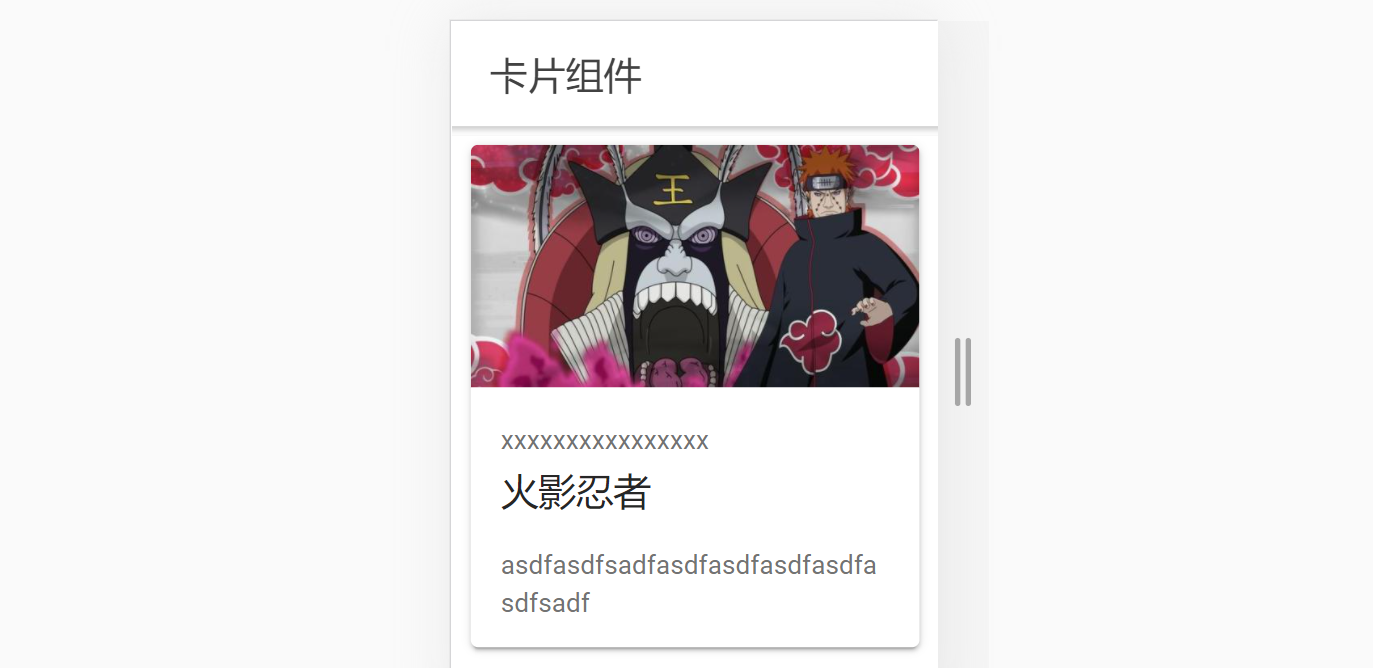
https://ionicons.com/



<!-- 图标 -->
  
<ion-app>
  
 <ion-header>
  
 <ion-toolbar>
  
 <ion-title>图标</ion-title>
  
 </ion-toolbar>
  
 </ion-header>
  
   
 <ion-content>
  
 <!-- https://ionicons.com/ -->
  
 <ion-icon
  
 name="logo-wechat"
  
 color="success"
  
 style="font-size: 3rem"
  
 ></ion-icon>
  
   
 <!-- 再找几个图标并展示: 支付宝, 小房子:首页, 心-喜欢, 齿轮-设置 -->
  
 <ion-button fill="clear">
  
 <ion-icon
  
 name="logo-alipay"
  
 color="primary"
  
 style="font-size: 2rem"
  
 ></ion-icon>
  
 </ion-button>
  
 </ion-content>
  
</ion-app>

### 卡片组件

https://ionicframework.com/docs/api/card



<!-- 卡片组件 -->
  
<ion-app>
  
 <ion-header>
  
 <ion-toolbar>
  
 <ion-title>卡片组件</ion-title>
  
 </ion-toolbar>
  
 </ion-header>
  
   
 <ion-content>
  
 <!-- i-card-full -->
  
 <ion-card>
  
 <img
  
 src="https://ss3.bdstatic.com/70cFv8Sh\_Q1YnxGkpoWK1HF6hhy/it/u=1315626315,169248210&fm=26&gp=0.jpg"
  
 alt=""
  
 />
  
 <ion-card-header>
  
 <ion-card-subtitle>xxxxxxxxxxxxxxxx</ion-card-subtitle>
  
 <ion-card-title>火影忍者</ion-card-title>
  
 </ion-card-header>
  
 <ion-card-content>
  
 asdfasdfsadfasdfasdfasdfasdfasdfsadf
  
 </ion-card-content>
  
 </ion-card>
  
 </ion-content>
  
</ion-app>

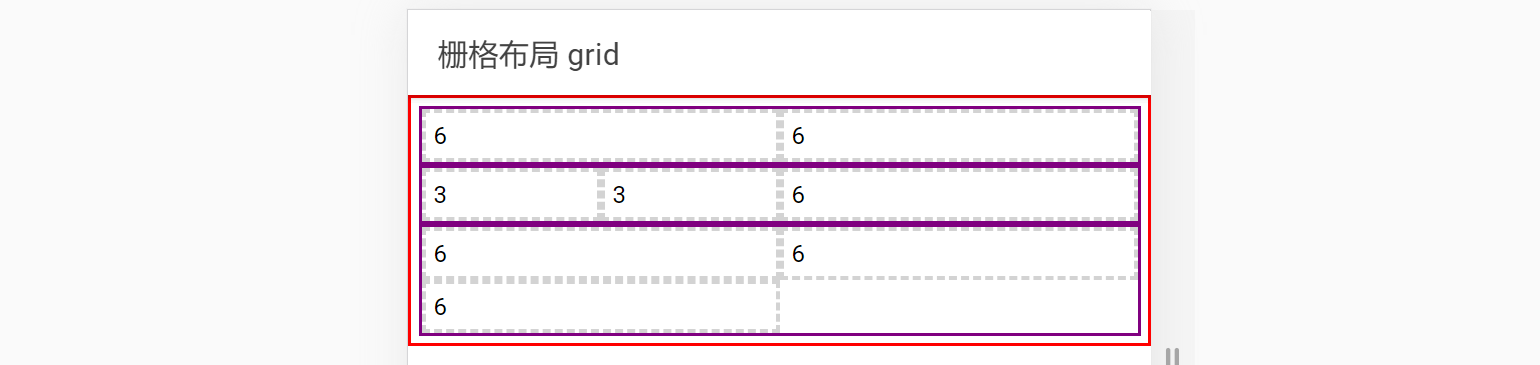
### 横向滚动

https://ionicframework.com/docs/api/slides

<!-- 横向滚动 -->
  
<ion-app>
  
 <ion-header>
  
 <ion-toolbar>
  
 <ion-title>横向滚动</ion-title>
  
 </ion-toolbar>
  
 </ion-header>
  
   
 <ion-content \*ngIf="res">
  
 <!-- 小程序 swiper 和 swiper-item -->
  
 <!-- angular: ion-slides 和 ion-slide -->
  
   
 <!-- 通过 options 属性可以添加 个性化的配置 -->
  
 <!-- autoplay:true 自动滚动 -->
  
 <!-- delay: 时间间隔 -->
  
 <!-- 默认设定: 手动操作之后会停止 自动滚动
  
 disableOnInteraction: 当操作之后是否停止自动滚动
  
 -->
  
 <!-- loop: 循环滚动 -->
  
 <!-- pager: 页数指示点 -->
  
 <ion-slides
  
 [options]="{
  
 autoplay: { delay: 1000, disableOnInteraction: false },
  
 loop: true
  
 }"
  
 pager
  
 >
  
 <!-- slide: 就是滚动项 -->
  
 <ion-slide \*ngFor="let item of res.result">
  
 <ion-card>
  
 <img [src]="item.img" alt="" />
  
 </ion-card>
  
 </ion-slide>
  
 </ion-slides>
  
 </ion-content>
  
</ion-app>

### 栅格布局 Grid

https://ionicframework.com/docs/api/grid



<!-- 栅格布局 -->
  
<ion-app>
  
 <ion-header>
  
 <ion-toolbar>
  
 <ion-title>栅格布局 grid</ion-title>
  
 </ion-toolbar>
  
 </ion-header>
  
   
 <ion-content>
  
 <!-- 把宽度等分成 12 列 -->
  
 <!-- i-grid -->
  
 <ion-grid fixed>
  
 <ion-row>
  
 <ion-col size="6">6</ion-col>
  
 <ion-col size="6">6</ion-col>
  
 </ion-row>
  
   
 <ion-row>
  
 <ion-col size="3">3</ion-col>
  
 <ion-col size="3">3</ion-col>
  
 <ion-col size="6">6</ion-col>
  
 </ion-row>
  
   
 <!-- 1行 12列, 超出12. 放不下的会自动折行 -->
  
 <ion-row>
  
 <ion-col size="6">6</ion-col>
  
 <ion-col size="6">6</ion-col>
  
 <ion-col size="6">6</ion-col>
  
 </ion-row>
  
 </ion-grid>
  
 </ion-content>
  
</ion-app>

### 加载更多

https://ionicframework.com/docs/api/infinite-scroll

import { Component } from "@angular/core";
  
   
import { Platform } from "@ionic/angular";
  
import { SplashScreen } from "@ionic-native/splash-screen/ngx";
  
import { StatusBar } from "@ionic-native/status-bar/ngx";
  
import { HttpClient } from "@angular/common/http";
  
import { url } from "inspector";
  
   
@Component({
  
 selector: "app-root",
  
 templateUrl: "app.component.html",
  
 styleUrls: ["app.component.scss"],
  
})
  
export class AppComponent {
  
 constructor(
  
 private platform: Platform,
  
 private splashScreen: SplashScreen,
  
 private statusBar: StatusBar,
  
 private http: HttpClient
  
 ) {
  
 this.initializeApp();
  
 }
  
   
 res: MeiTu;
  
   
 // 下拉刷新
  
 doRefresh(event) {
  
 let url = "https://api.apiopen.top/getImages?page=7";
  
   
 this.http.get(url).subscribe((res: MeiTu) => {
  
 console.log(res);
  
 // 新的数据 覆盖 旧的
  
 this.res = res;
  
   
 this.page = 7; //页数回归初值
  
 event.target.complete(); // 让下拉刷新状态结束
  
 });
  
 }
  
   
 // 触底 加载更多触发
  
 page = 7; //当前页
  
   
 loadData(event) {
  
 let url = "https://api.apiopen.top/getImages?page=" + (this.page + 1);
  
   
 this.http.get(url).subscribe((res: MeiTu) => {
  
 console.log(res);
  
 // 合并数据: 旧数据 和 新数据 合并
  
 res.result = this.res.result.concat(res.result);
  
   
 this.res = res;
  
 // 告诉组件: 本次加载更多操作已结束. 这样组件才可以准备下一次
  
 event.target.complete();
  
   
 this.page++; //页数更新
  
 });
  
 }
  
   
 ngOnInit(): void {
  
 let url = "https://api.apiopen.top/getImages?page=7";
  
   
 this.http.get(url).subscribe((res: MeiTu) => {
  
 console.log(res);
  
   
 this.res = res;
  
 });
  
 }
  
   
 initializeApp() {
  
 this.platform.ready().then(() => {
  
 this.statusBar.styleDefault();
  
 this.splashScreen.hide();
  
 });
  
 }
  
}
  
   
////////////////////////////////////////
  
interface MeiTu {
  
 code: number;
  
 message: string;
  
 result: Result[];
  
}
  
   
interface Result {
  
 id: number;
  
 time: string;
  
 img: string;
  
}

<ion-app>
  
 <ion-header>
  
 <ion-toolbar>
  
 <ion-title>多列布局</ion-title>
  
 </ion-toolbar>
  
 </ion-header>
  
   
 <ion-content \*ngIf="res">
  
 <!-- 下拉刷新 -->
  
 <!-- i-refresher -->
  
 <ion-refresher
  
 slot="fixed"
  
 (ionRefresh)="doRefresh($event)"
  
 pullFactor="0.8"
  
 pullMin="60"
  
 pullMax="120"
  
 >
  
 <ion-refresher-content></ion-refresher-content>
  
 </ion-refresher>
  
   
 <ion-grid fixed>
  
 <ion-row>
  
 <!-- ion-col 自带 5px 内边距 -->
  
 <ion-col size="6" \*ngFor="let item of res.result">
  
 <!-- 卡片自带 10px 外边距 -->
  
 <ion-card style="margin: 0">
  
 <img [src]="item.img" alt="" />
  
 </ion-card>
  
 </ion-col>
  
 </ion-row>
  
 </ion-grid>
  
   
 <!-- 加载更多 -->
  
 <!-- i-infinite-scroll -->
  
 <ion-infinite-scroll
  
 threshold="25%"
  
 [disabled]="false"
  
 position="bottom"
  
 (ionInfinite)="loadData($event)"
  
 >
  
 <ion-infinite-scroll-content
  
 loadingSpinner="lines"
  
 loadingText="涛哥,不要急..."
  
 >
  
 </ion-infinite-scroll-content>
  
 </ion-infinite-scroll>
  
 </ion-content>
  
</ion-app>

## 作业

接口地址: https://api.apiopen.top/getWangYiNews?page=1

网易新闻练习

* 两列布局
* 有下拉刷新 和 加载更多

