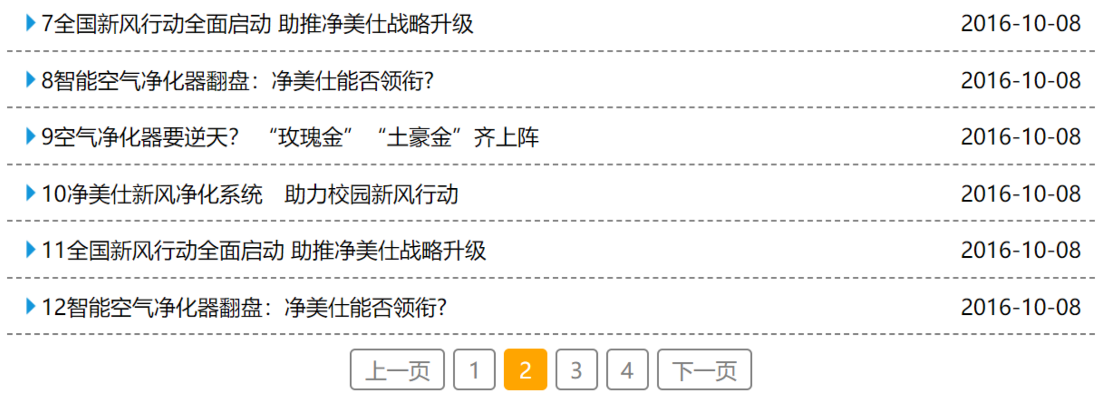
Angular05

今日内容:

- 完成 angular 作业， 需要angular的项目包
- 学子 ionic 组件, 需要 ionic 项目包



7全国新风行动全面启动 助推净美仕战略升级 2016-10-08
8智能空气净化器翻盘：净美仕能否领衔? 2016-10-08
9空气净化器要逆天？ “玫瑰金” “土豪金”齐上阵 2016-10-08
10净美仕新风净化系统 助力校园新风行动 2016-10-08
11全国新风行动全面启动 助推净美仕战略升级 2016-10-08
12智能空气净化器翻盘：净美仕能否领衔? 2016-10-08

```html
<p>news works!</p>

<div class="news" *ngIf="res">
  <div class="content">
    <div *ngFor="let item of res.data" class="cell">
      <img src="/assets/right.png" alt="" />
      <span>{{ item.title }}</span>
      <span class="cell-time">{{ item.pubTime|date:'yyyy-MM-dd' }}</span>
    </div>
  </div>
  <div class="pages">
    <span class="page" (click)="getData(res.pageNum-1)" *ngIf="res.pageNum>1"
      >上一页</span
    >
    <span class="page-disabled" *ngIf="res.pageNum==1">上一页</span>
    <!-- angular: 不支持遍历数字;   而vue支持 v-for="item in 4" -->
    <span
      class="page"
      *ngFor="let item of range(res.pageCount)"
      [ngClass]="{'cur': item==res.pageNum}"
      (click)="getData(item)"
      >{{item}}</span
    >
    <span
      class="page"
      (click)="getData(res.pageNum+1)"
      *ngIf="res.pageNum<res.pageCount"
      >下一页</span
    >
    <span class="page-disabled" *ngIf="res.pageNum==res.pageCount">下一页</span>
  </div>
</div>
```

```css
.news {
  width: 800px;
  margin: 0 auto;
}

.cell {
  padding: 10px;
  display: flex;
  border-bottom: 1px dashed gray;
  align-items: center;
}

.cell > img {
  width: 15px;
  height: 15px;
}

.cell-time {
  margin-left: auto;
}

.pages {
  text-align: center;
  margin-top: 10px;
  user-select: none;
}

.page-disabled {
  display: inline-block;
  padding: 4px 10px;
  border-radius: 4px;
  border: 1px solid lightgray;
  color: lightgray;
  margin: 0 3px;
}

.page {
  display: inline-block;
  padding: 4px 10px;
  border-radius: 4px;
  border: 1px solid gray;
  color: gray;
  margin: 0 3px;
}

.page:hover {
  border-color: orange;
  color: white;
  background-color: orange;
  cursor: pointer;
}

.cur {
  border-color: orange;
  color: white;
  background-color: orange;
}
```

```typescript
import { HttpClient } from "@angular/common/http";
import { Component, OnInit } from "@angular/core";

@Component({
  selector: "app-news",
  templateUrl: "./news.component.html",
  styleUrls: ["./news.component.css"],
})
export class NewsComponent implements OnInit {
  // 依赖注入机制:
  // 构造方法是实例化时触发, 其中的参数必须传递
  // 声明依赖:  语法糖写法  权限词 参数名: 参数类型
  // 系统注入: 系统会自动注入指定类型的值
  constructor(public http: HttpClient) {}

  res: Result; //属性才能跨方法使用 并 在html中使用

  // 生命周期: 组件加载中
  ngOnInit(): void {
    this.getData(1);
  }

  getData(pno) {
    let url =
      "http://101.96.128.94:9999/mfresh/data/news_select.php?pageNum=" + pno;

    this.http.get(url).subscribe((res: Result) => {
      console.log(res);

      //局部变量 保存到属性
      this.res = res;
    });
  }

  // 制作函数: 4 -> [1,2,3,4]
  range(num) {
    let arr = [];

    for (let i = 1; i <= num; i++) {
      arr.push(i);
    }

    return arr;
  }
}

interface Result {
  data: ResultData[];
  pageCount: number;
  pageNum: number;
  pageSize: number;
  totalRecord: number;
}
```

```
interface ResultData {
  content: string;
  nid: string;
  pubTime: string;
  title: string;
}
```

ionic

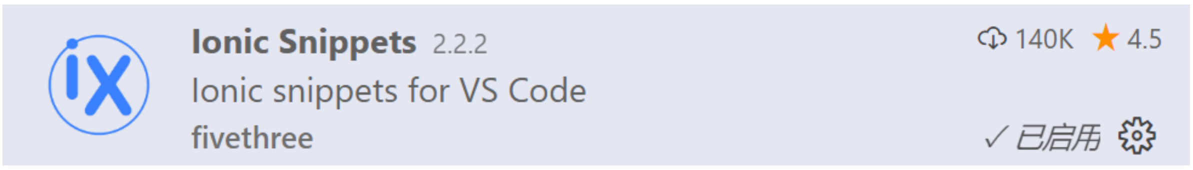
组件库: https://ionicframework.com/docs/components

> 本质上, 就是一些自定义的组件, 带有固定的手机端样式css

ionic命令回顾:

```
脚手架安装
npm i -g @ionic/cli

生成项目包
ionic start 包名 包类型(blank|tabs|sidemenu)

运行:
* ionic s      端口号8100
* ng s -o      端口号4200
```

插件


Ionic Snippets 2.2.2

Ionic snippets for VS Code

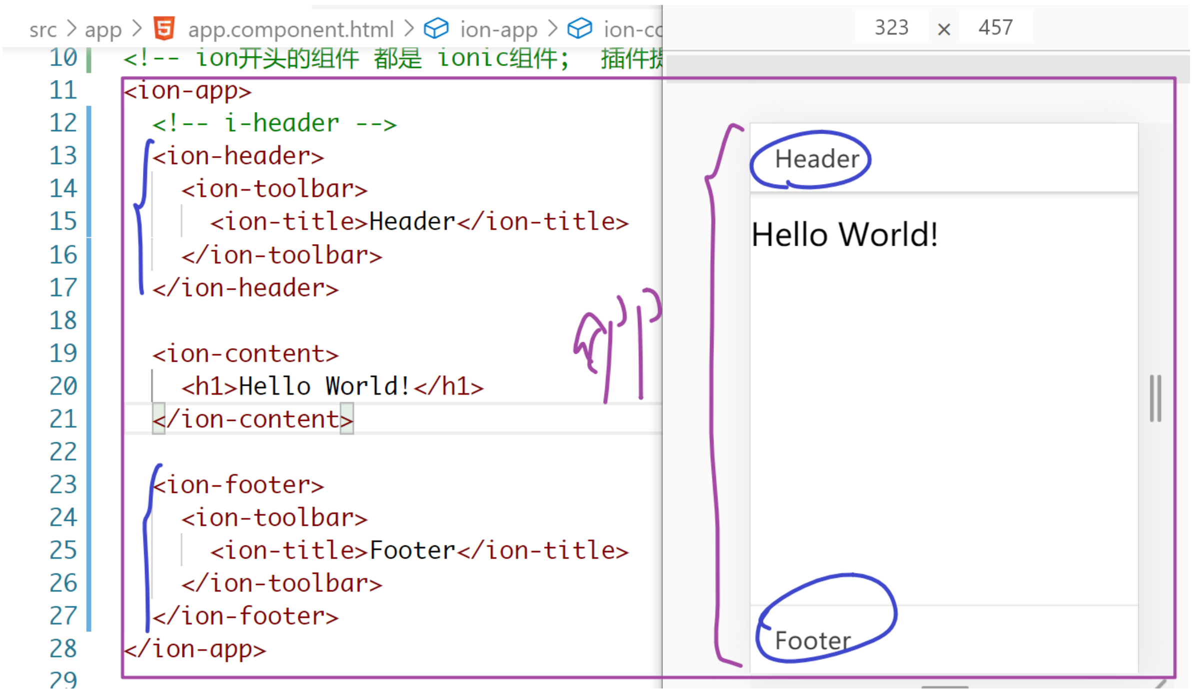
fivethree

140K ★ 4.5

✓ 已启用

容器

```
 容器:
  整个的容器: <ion-app>  类似web的 <html>
  页面分: 头 身体 尾
  * 头: ion-header
  * 身体: ion-content
  * 尾: ion-footer
```



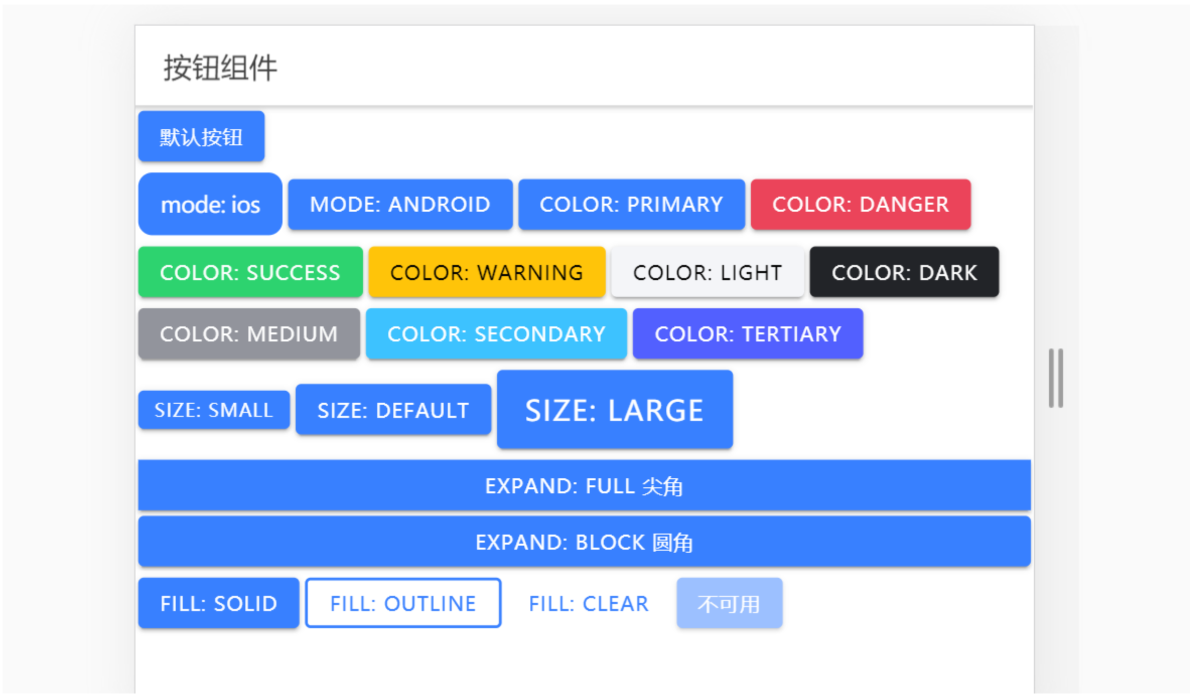
src > app > app.component.html > ion-app > ion-c

323 × 457

```
<!-- ion开头的组件 都是 ionic组件;  插件
<ion-app>
  <!-- i-header -->
  <ion-header>
    <ion-toolbar>
      <ion-title>Header</ion-title>
    </ion-toolbar>
  </ion-header>

  <ion-content>
    <h1>Hello World!</h1>
  </ion-content>

  <ion-footer>
    <ion-toolbar>
      <ion-title>Footer</ion-title>
    </ion-toolbar>
  </ion-footer>
</ion-app>
```

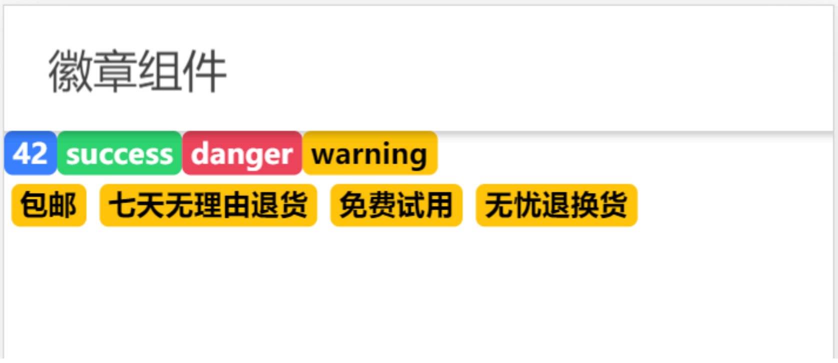
按钮组件

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



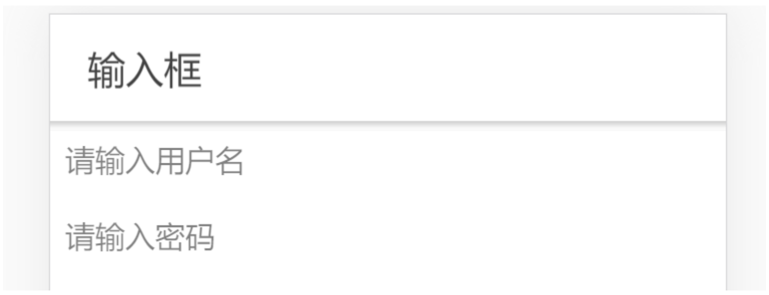
```
<!-- 按钮组件: 本质上就是div标签 然后添加了很多样式,  最后看起来就是手机端的组件样子 -->
<ion-app>
  <ion-header>
    <ion-toolbar>
      <ion-title>按钮组件</ion-title>
    </ion-toolbar>
  </ion-header>

  <ion-content>
    <!-- ionic组件都具备两种风格: ios 和 Android, 默认会自动随环境变化 -->
    <ion-button>默认按钮</ion-button>
    <br />
    <!-- 每个组件 官方提供了很多属性, 可以定制样式 -->
    <ion-button mode="ios">mode: ios</ion-button>
    <!-- Android是 md风格 -->
    <ion-button mode="md">mode: android</ion-button>

    <!-- 颜色属性: ionic官方提供了一些默认的风格主题 -->
    <ion-button color="primary">color: primary</ion-button>
    <ion-button color="danger">color: danger</ion-button>
    <ion-button color="success">color: success</ion-button>
    <ion-button color="warning">color: warning</ion-button>
    <ion-button color="light">color: light</ion-button>
    <ion-button color="dark">color: dark</ion-button>
    <ion-button color="medium">color: medium</ion-button>
    <ion-button color="secondary">color: secondary</ion-button>
    <ion-button color="tertiary">color: tertiary</ion-button>

    <!-- 按钮大小 -->
    <br />
    <ion-button size="small">size: small</ion-button>
    <ion-button size="default">size: default</ion-button>
    <ion-button size="large">size: large</ion-button>

    <!-- 按钮变块元素 -->
    <ion-button expand="full">expand: full 尖角</ion-button>
    <ion-button expand="block">expand: block 圆角</ion-button>

    <!-- 按钮的填充 solid实心 outline空心 clear无边框无背色-->
    <ion-button fill="solid">fill: solid</ion-button>
    <ion-button fill="outline">fill: outline</ion-button>
    <ion-button fill="clear">fill: clear</ion-button>

    <!-- 不可用 -->
    <ion-button disabled>不可用</ion-button>

    <!-- 点击事件: (click)="" -->
  </ion-content>
</ion-app>
```

徽记组件



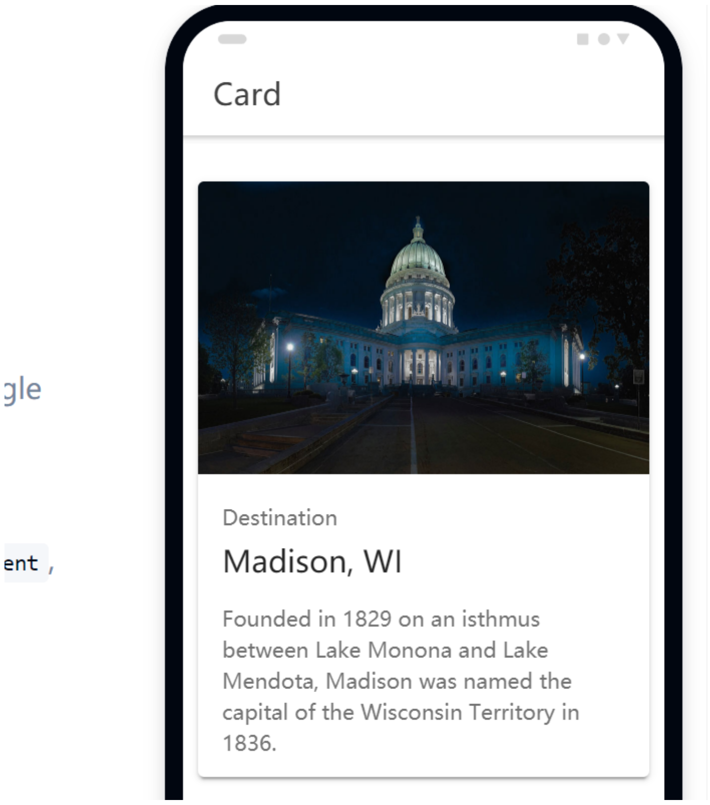
```
    <ion-badge color="primary">42</ion-badge>
    <ion-badge color="success">success</ion-badge>
    <ion-badge color="danger">danger</ion-badge>
    <ion-badge color="warning">warning</ion-badge>

    <div>
      <ion-badge color="warning">包邮</ion-badge>
      <ion-badge color="warning">七天无理由退货</ion-badge>
      <ion-badge color="warning">免费试用</ion-badge>
      <ion-badge color="warning">无忧退换货</ion-badge>
    </div>
```

输入框组件



```
  <ion-content>
    <ion-input type="text" placeholder="请输入用户名"></ion-input>
    <ion-input type="password" placeholder="请输入密码"></ion-input>
  </ion-content>
```

卡片组件



Destination

Madison, WI

Founded in 1829 on an isthmus between Lake Monona and Lake Mendota, Madison was named the capital of the Wisconsin Territory in 1836.

```
<!-- 卡片 -->
<ion-app>
  <ion-header>
    <ion-toolbar>
      <ion-title>卡片组件</ion-title>
    </ion-toolbar>
  </ion-header>

  <ion-content>
    <ion-card>
      <img

 src="http://5b0988e595225.cdn.sohucs.com/images/20190107/626de8aa4ba7407597d623
51116f8260.jpeg"
        alt=""
      />

      <!-- ion-card-header 本质就是 <div style="padding:16px"> -->
      <ion-card-header>
        <ion-card-subtitle>Awesome Subtitle</ion-card-subtitle>
        <ion-card-title>Awesome Title</ion-card-title>
      </ion-card-header>

      <ion-card-content>
        Awesome contentAwesome contentAwesome contentAwesome contentAwesome
        contentAwesome contentAwesome contentAwesome content
      </ion-card-content>
    </ion-card>
  </ion-content>
</ion-app>
```

卡片练习:

> 网络请求 必须在 app.module.ts 中注入 网络模块

```
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";

@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();
  }

  initializeApp() {
    this.platform.ready().then(() => {
      this.statusBar.styleDefault();
      this.splashScreen.hide();
    });
  }

  res: Result; //属性才能在html中使用

  // 生命周期: 挂载时
  ngOnInit(): void {
    let url = "https://api.apiopen.top/getImages";

    this.http.get(url).subscribe((res: Result) => {
      console.log(res);

      this.res = res;
    });
  }
}

// 声明返回值的类型, 这样html中使用才有代码提示!
interface Result {
  code: number;
  message: string;
  result: ResultData[]; // 数组类型, 中的元素都是 ResultData 类型
}

interface ResultData {
  id: number;
  img: string;
  time: string;
}
```

```
<!-- 卡片 -->
<ion-app>
  <ion-header>
    <ion-toolbar>
      <ion-title>美图秀秀</ion-title>
    </ion-toolbar>
  </ion-header>

  <ion-content *ngIf="res">
    <ion-card *ngFor="let item of res.result">
      <img [src]="item.img" alt="" />
    </ion-card>
  </ion-content>
</ion-app>
```

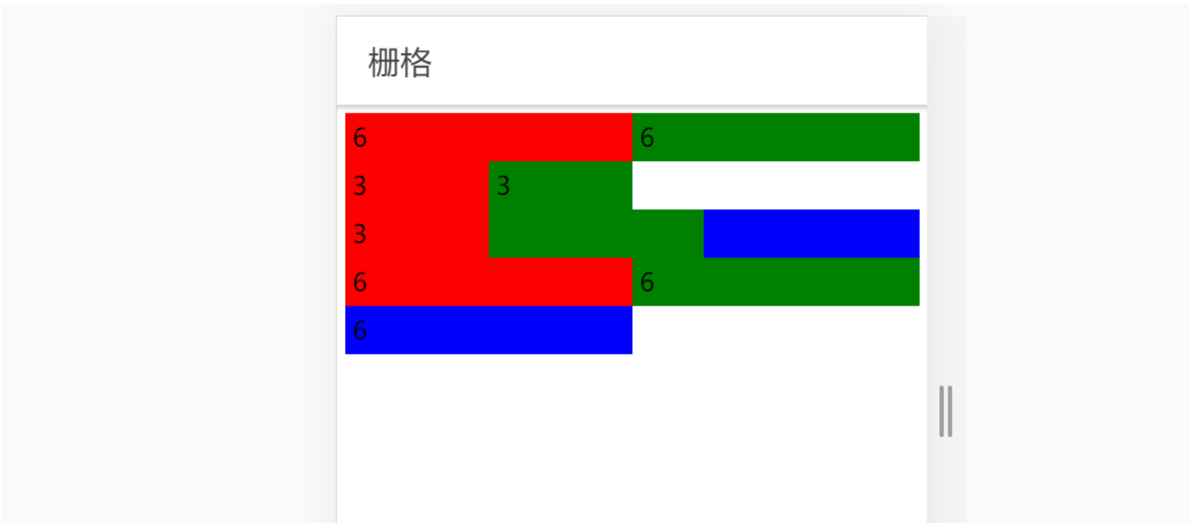
横向滚动组件

> 小程序中: swiper swiper-item
>
> vue中: 第三方的 swiper 扩展
>
> angular中: 基于 swiper 扩展制作的 slides 和 slide

```html
<!-- 卡片 -->
<ion-app>
  <ion-header>
    <ion-toolbar>
      <ion-title>横向滚动</ion-title>
    </ion-toolbar>
  </ion-header>

  <ion-content *ngIf="res">
    <!-- slides外层容器， slide每一个滚动项目 -->
    <!--
      pager: 页数指示点
      options: 可以个性化进行各种配置，具体要看官方文档:
          * autoplay: true   开启自动滚动，使用默认配置
            - delay: 滚动间隔，单位ms
            - disableOnInteraction: 用户触摸后,停止自动滚动. 默认为true 代表开启此配置
          * loop: 环路，滚动循环

     -->
    <ion-slides
      pager
      [options]="{autoplay: {delay:1000, disableOnInteraction:false},
loop:true}"
    >
      <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

> 之前学过的典型的栅格布局有两种
>
> - table
> - Bootstrap: 固定分12份


栅格

6 6
3 3
3
6 6
6

```
<ion-app>
  <ion-header>
    <ion-toolbar>
      <ion-title>栅格</ion-title>
    </ion-toolbar>
  </ion-header>

  <ion-content>
    <ion-grid fixed>
      <!-- row:行 col:列 -->
      <ion-row>
        <!-- size:  最大12 -->
        <ion-col size="6" style="background-color: red;">6</ion-col>
        <ion-col size="6" style="background-color: green;">6</ion-col>
      </ion-row>

      <ion-row>
        <ion-col size="3" style="background-color: red;">3</ion-col>
        <ion-col size="3" style="background-color: green;">3</ion-col>
      </ion-row>

      <ion-row>
        <ion-col size="3" style="background-color: red;">3</ion-col>
        <!-- 剩余空间,被没有size属性的平分 -->
        <ion-col style="background-color: green;"></ion-col>
        <ion-col style="background-color: blue;"></ion-col>
      </ion-row>

      <!-- 一行共12列, 如果超出12, 则自动折行显示 -->
      <ion-row>
        <ion-col size="6" style="background-color: red;">6</ion-col>
        <ion-col size="6" style="background-color: green;">6</ion-col>
        <ion-col size="6" style="background-color: blue;">6</ion-col>
      </ion-row>
    </ion-grid>
  </ion-content>
</ion-app>
```

```
<ion-app>
  <ion-header>
    <ion-toolbar>
      <ion-title>美图秀秀</ion-title>
    </ion-toolbar>
  </ion-header>

  <ion-content *ngIf="res">
    <ion-grid fixed>
      <ion-row>
        <ion-col size="6" *ngFor="let item of res.result">
          <!-- card: 卡片自带圆角和阴影, 好看! 自带外边距 -->
          <ion-card style="margin: 0;">
            <img [src]="item.img" alt="" />
          </ion-card>
        </ion-col>
      </ion-row>
    </ion-grid>
  </ion-content>
</ion-app>
```

下拉刷新和加载更多

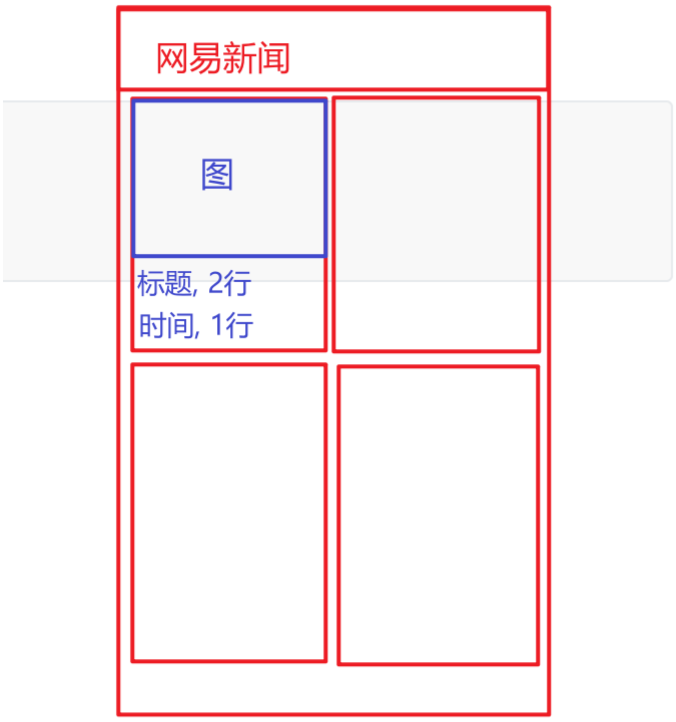
```
import { Component } from "@angular/core";

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";

@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();
  }

  initializeApp() {
    this.platform.ready().then(() => {
      this.statusBar.styleDefault();
      this.splashScreen.hide();
    });
  }

  res: Result; //属性才能在html中使用
  // 生命周期: 挂载时
  ngOnInit(): void {
    let url = this.url + 7;

    this.http.get(url).subscribe((res: Result) => {
      console.log(res);

      this.res = res;
    });
  }

  pno = 7; //当前页, 默认7.  经测试, 从第7页之后的图都有
  // 下拉刷新,加载更多,首页 都要用此变量;   属性才能复用
  url = "https://api.apiopen.top/getImages?count=8&page=";

  // 子父传参: 加载更多组件出现时,自动触发此方法
  loadData(e) {
    // console.log(e); //e就是加载更多组件

    let nextPage = this.pno + 1; //下一页=当前页+1
    let url = this.url + nextPage;

    this.http.get(url).subscribe((res: Result) => {
      console.log(res);

      // 新的数据添加到旧的数据中
      // concat(): 用于拼接两个数组, 返回新的大数组
      res.result = this.res.result.concat(res.result);

      this.res = res;

      // 页数改
      this.pno = nextPage;
      // 告诉组件,本次加载更多操作已结束, 可以开始下一次了
      e.target.complete(); //complete: 完成
    });
  }

  //下拉刷新
  doRefresh(e) {
    let url = this.url + 7;

    this.http.get(url).subscribe((res: Result) => {
      this.res = res; //新数据覆盖旧的, 就是刷新

      //重置当前页
      this.pno = 7;

      //结束下拉刷新动画
      e.target.complete();
    });
  }
}

// 声明返回值的类型, 这样html中使用才有代码提示!
interface Result {
  code: number;
  message: string;
  result: ResultData[]; // 数组类型，中的元素都是 ResultData 类型
}

interface ResultData {
  id: number;
  img: string;
  time: string;
}
```

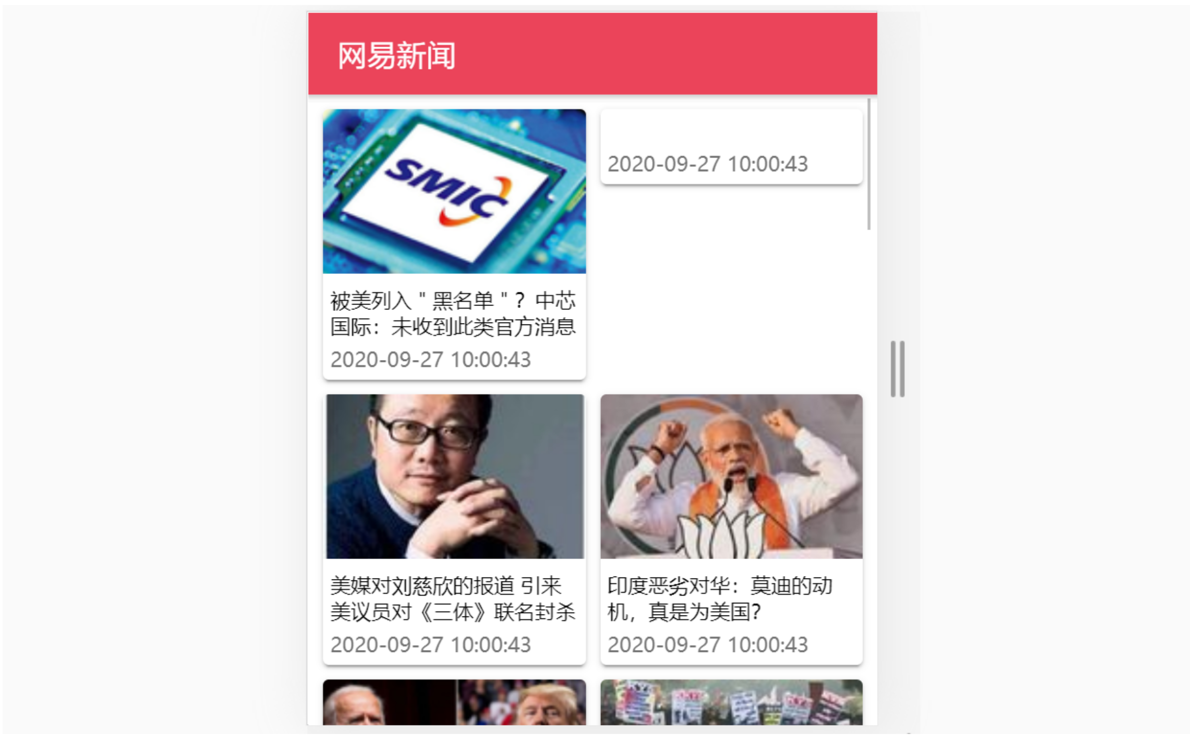
```
<ion-app>
  <ion-header>
    <ion-toolbar>
      <ion-title>美图秀秀</ion-title>
    </ion-toolbar>
  </ion-header>

  <ion-content *ngIf="res">
    <!-- ire -->
    <ion-refresher slot="fixed" (ionRefresh)="doRefresh($event)">
      <ion-refresher-content></ion-refresher-content>
    </ion-refresher>

    <ion-grid fixed>
      <ion-row>
        <ion-col size="6" *ngFor="let item of res.result">
          <!-- card: 卡片自带圆角和阴影，好看！ 自带外边距 -->
          <ion-card style="margin: 0;">
            <img [src]="item.img" alt="" />
          </ion-card>
        </ion-col>
      </ion-row>
    </ion-grid>

    <!-- iin -->
    <ion-infinite-scroll
      threshold="25%"
      [disabled]="false"
      position="bottom"
      (ionInfinite)="loadData($event)"
    >
      <ion-infinite-scroll-content
        loadingSpinner="dots"
        loadingText="加载中，请稍后..."
      >
      </ion-infinite-scroll-content>
    </ion-infinite-scroll>
  </ion-content>
</ion-app>
```

练习

接口地址:

```
https://api.apiopen.top/getWangYiNews?page=1

参数: page 代表页数, 从1开始
```





2020-09-27 10:00:43

被美列入＂黑名单＂? 中芯国际: 未收到此类官方消息
2020-09-27 10:00:43

美媒对刘慈欣的报道 引来美议员对《三体》联名封杀
2020-09-27 10:00:43

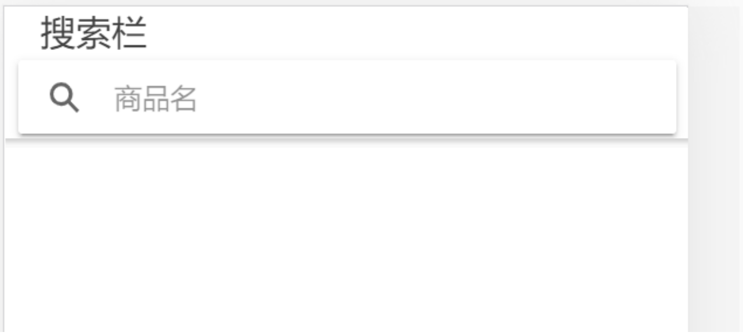
印度恶劣对华: 莫迪的动机, 真是为美国?
2020-09-27 10:00:43

```
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";

@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();
  }

  initializeApp() {
    this.platform.ready().then(() => {
      this.statusBar.styleDefault();
      this.splashScreen.hide();
    });
  }

  res: Result;

  ngOnInit(): void {
    let url = "https://api.apiopen.top/getWangYiNews?page=1";

    this.http.get(url).subscribe((res: Result) => {
      console.log(res);

      this.res = res;
    });
  }

  //当前页
  pno = 1;
  url = "https://api.apiopen.top/getWangYiNews?page=";

  //加载更多
  loadData(e) {
    let nextP = this.pno + 1;

    this.http.get(this.url + nextP).subscribe((res: Result) => {
      console.log(res);
      //合并数组 -> 更新数据 -> 更新页数 -> 结束加载更多
      res.result = this.res.result.concat(res.result);
      this.res = res;

      this.pno = nextP;

      e.target.complete();
    });
  }
  //下拉刷新
  doRefresh(e) {
    this.http.get(this.url + 1).subscribe((res: Result) => {
      console.log(res);

      this.res = res;
      this.pno = 1;
      e.target.complete();
    });
  }
}

interface Result {
  code: number;
  message: string;
  result: ResultData[]; // Array<any>  any[]: 数组类型中的元素是 any
}

interface ResultData {
  image: string;
  passtime: string;
  path: string;
  title: string;
}
```

```
<ion-app>
  <ion-header>
    <ion-toolbar color="danger">
      <ion-title>网易新闻</ion-title>
    </ion-toolbar>
  </ion-header>

  <ion-content *ngIf="res">
    <!-- 下拉刷新 -->
    <ion-refresher slot="fixed" (ionRefresh)="doRefresh($event)">
      <ion-refresher-content></ion-refresher-content>
    </ion-refresher>

    <ion-grid fixed>
      <ion-row>
        <ion-col size="6" *ngFor="let item of res.result">
          <ion-card>
            <img [src]="item.image" alt="" />
            <div class="box">
              <span>{{item.title}}</span>
              <span>{{item.passtime}}</span>
            </div>
          </ion-card>
        </ion-col>
      </ion-row>
    </ion-grid>

    <!-- iin -->
    <ion-infinite-scroll
```

```
      threshold="25%"
      position="bottom"
      (ionInfinite)="loadData($event)"
    >
      <ion-infinite-scroll-content
        loadingSpinner="bubbles"
        loadingText="加载中..."
      >
      </ion-infinite-scroll-content>
    </ion-infinite-scroll>
  </ion-content>
</ion-app>
```

搜索框

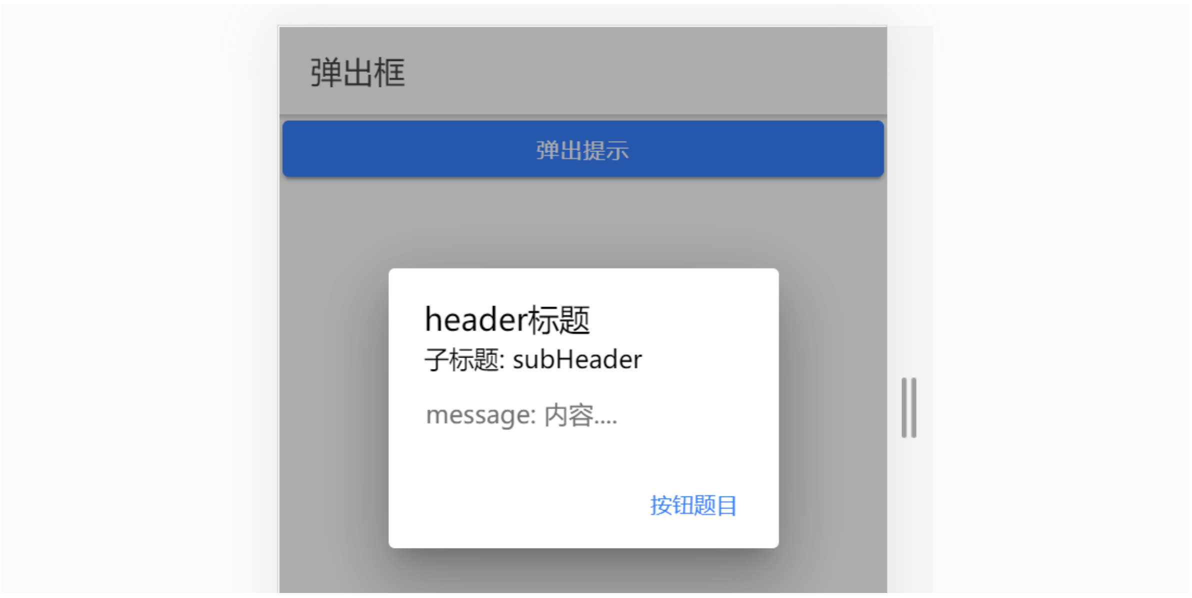

搜索栏

商品名

```
<!-- 搜索栏 -->
<ion-app>
  <ion-header>
    <ion-toolbar>
      <ion-title>搜索栏</ion-title>
      <ion-searchbar
        placeholder="商品名"
        (ionChange)="onSearchChange($event)"
      ></ion-searchbar>
    </ion-toolbar>
  </ion-header>

  <ion-content> </ion-content>
</ion-app>
```

```
  onSearchChange(e) {
    // console.log(e);
    let value = e.detail.value;
    console.log(value);

    //通常: 发送网络请求给服务器, 服务器查询出对应的结果反馈 并展示!
  }
```

弹出框组件



```
constructor(
    private alertC: AlertController
  ) {
    this.initializeApp();
  }
```

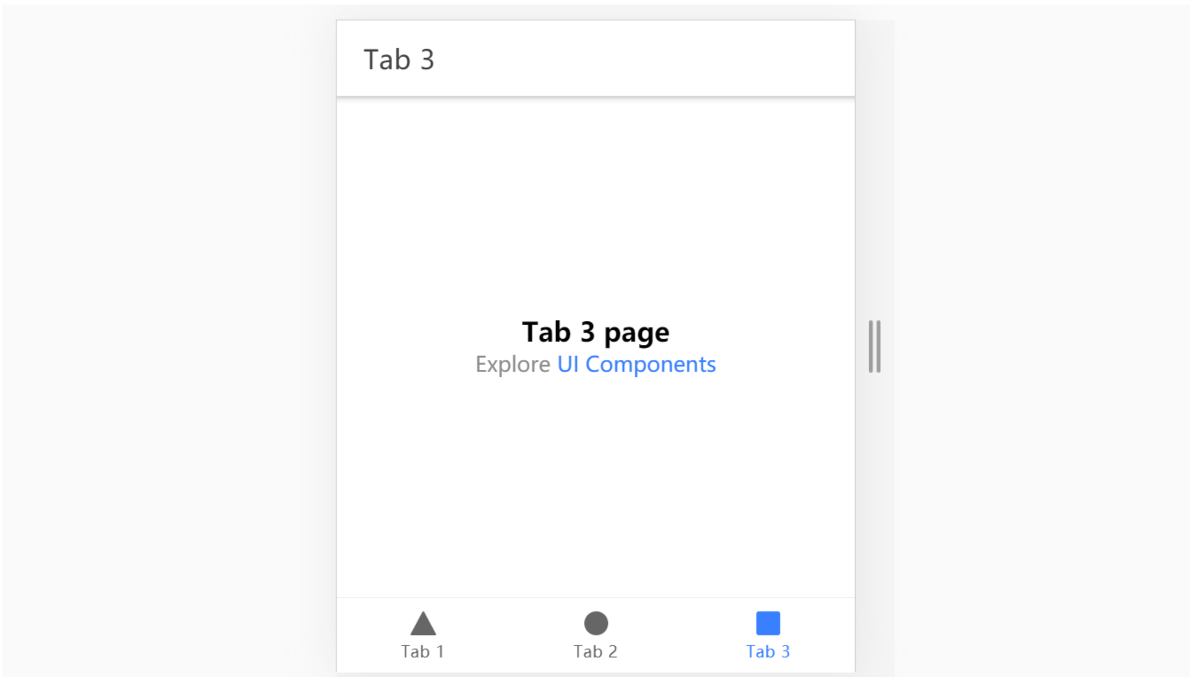
```
showAlert() {
    //弹出框, 需要使用 弹出框服务: 依赖注入 constructor
    //使用方式:  弹出框服务.创建(弹出框的配置).然后(弹出框=> 弹出框.弹出())
    this.alertC
      .create({
        header: "header标题",
        subHeader: "子标题: subHeader ",
        message: "message: 内容....",
        // buttons: ["OK", "Cancel"],
        buttons: [
          {
            text: "按钮题目",
            handler: () => {
              console.log("此按钮被点击时触发");
            },
          },
        ],
      })
      .then((res) => res.present());
  }
```

生成带有标签导航的包

包会出现在命令行执行所在目录下:

```
ionic start tabsApp tabs

* tabs 代表生成标签导航项目包

过程中的选项都回车
```

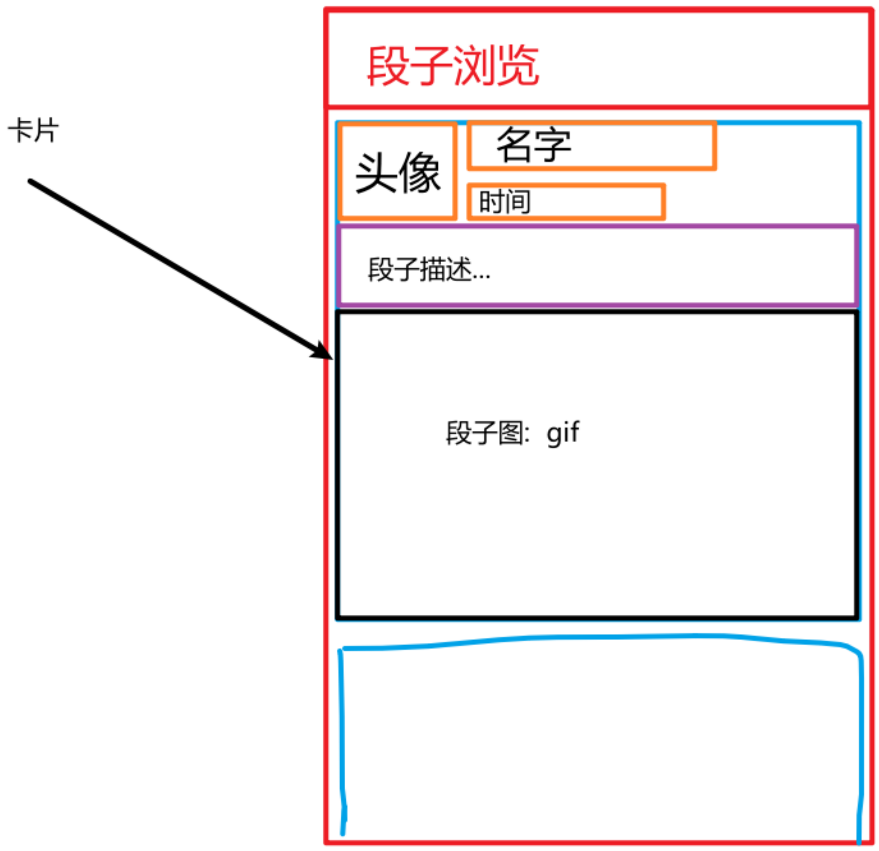
启动命令, 依然是: `ionic s`



作业

段子浏览

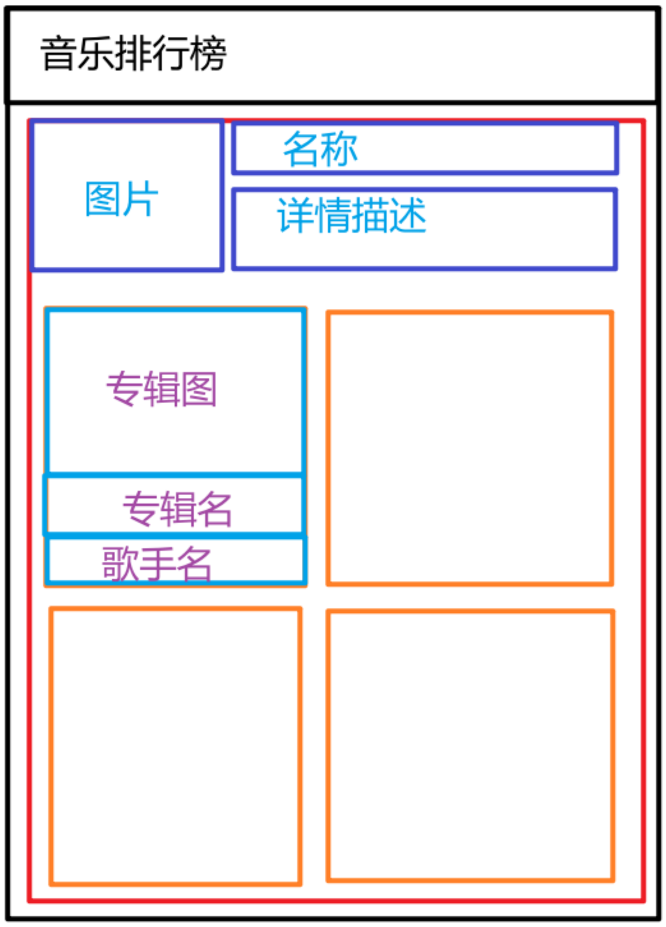
```
接口: https://api.apiopen.top/getJoke?count=5&type=gif&page=1
参数: page 代表页数, 从1开始
```



音乐排行榜

```
接口: https://api.apiopen.top/musicRankings
```

制作成横向滚动展示


音乐排行榜
图片
名称
详情描述
专辑图
专辑名
歌手名