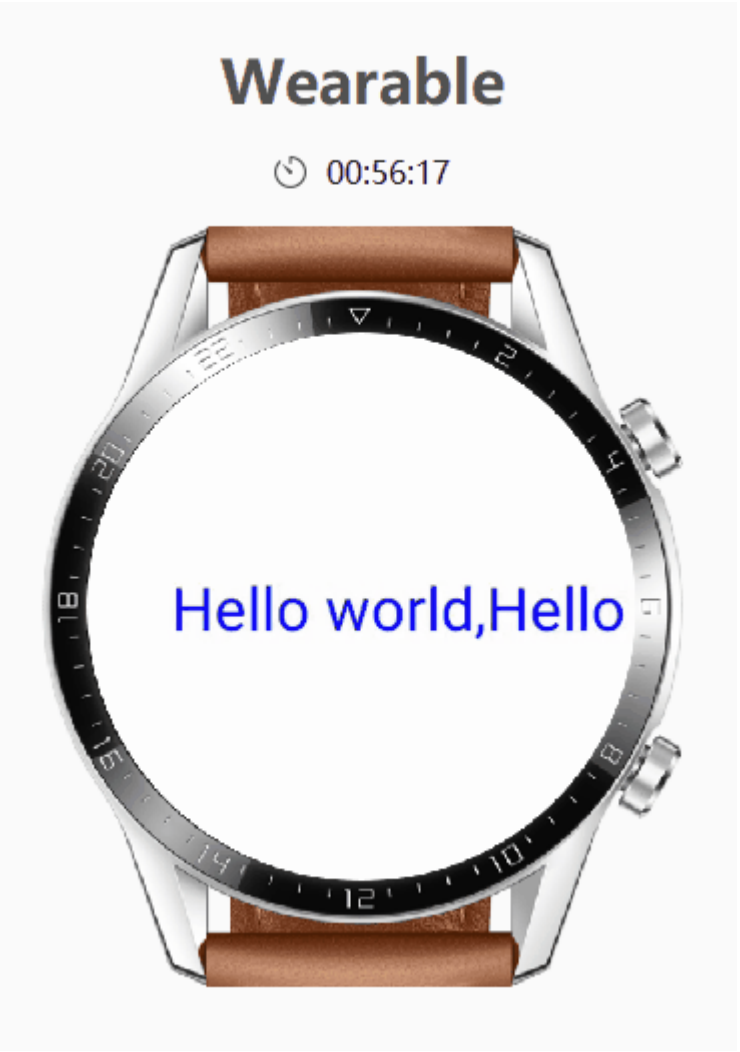
Harmony OS
入门系列课程 <快速上手>

快速掌握鸿蒙系统应用开发基础操作技巧



第 8 讲：JS-FA应用初体验

章节目录

课程目标

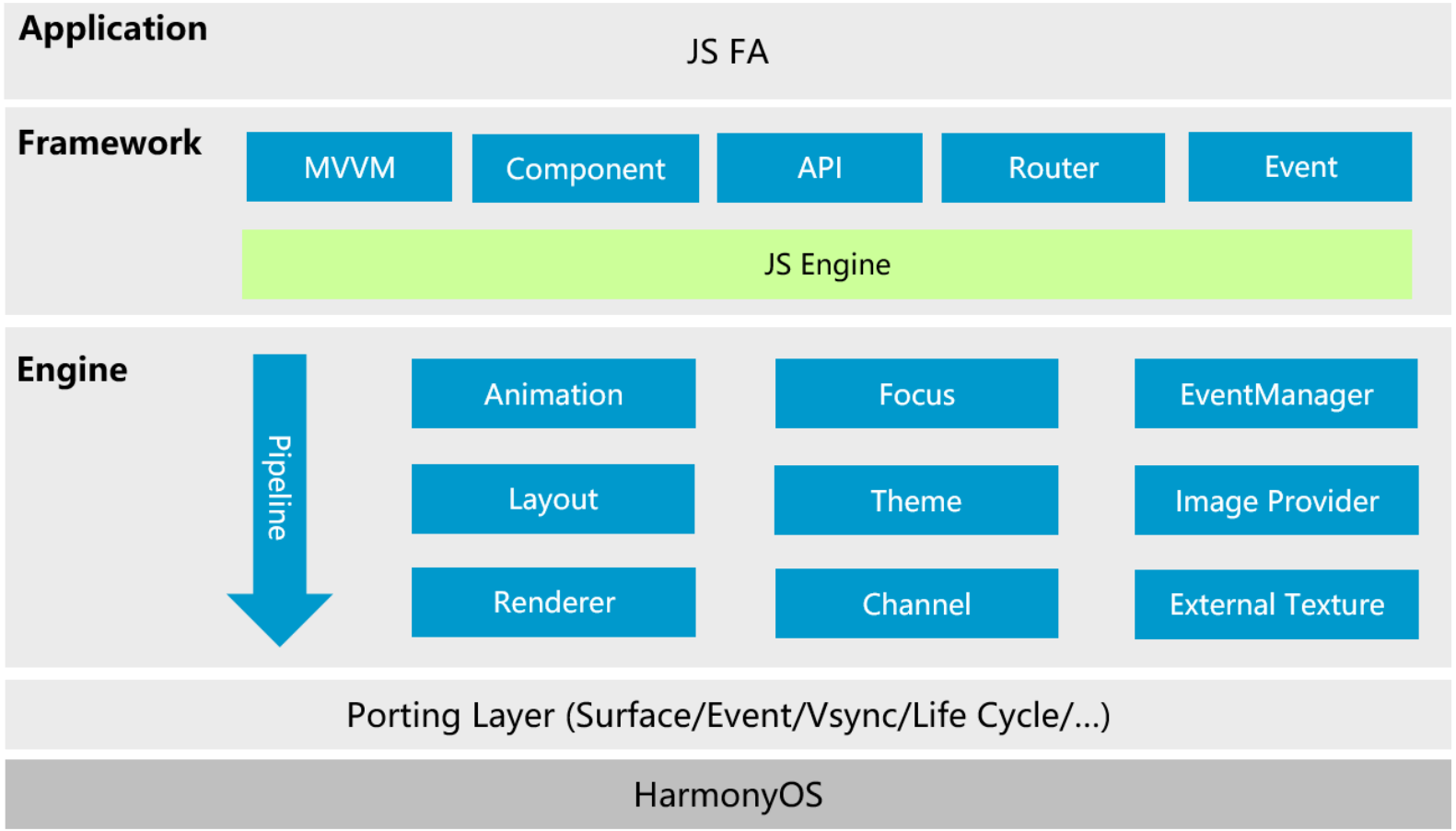
- 了解JS UI框架相关知识及架构；
- 掌握JS FA应用的目录结构；
- 掌握HML\CSS\JS的基础语法；
- 创建并开发JS FA应用；

JS FA概述

- JS UI框架是一种跨设备的高性能UI开发框架，支持声明式编程和跨设备多态UI。
- JS UI框架采用类HTML和CSS声明式编程语言作为页面布局和页面样式的开发语言，页面业务逻辑则支持ECMAScript规范的JavaScript语言。
- JS UI框架提供的声明式编程，可以让开发者避免编写UI状态切换的代码，视图配置信息更加直观。
- JS UI框架支持纯JavaScript、JavaScript和Java混合语言开发。JS FA指基于JavaScript或JavaScript和Java混合开发的FA。

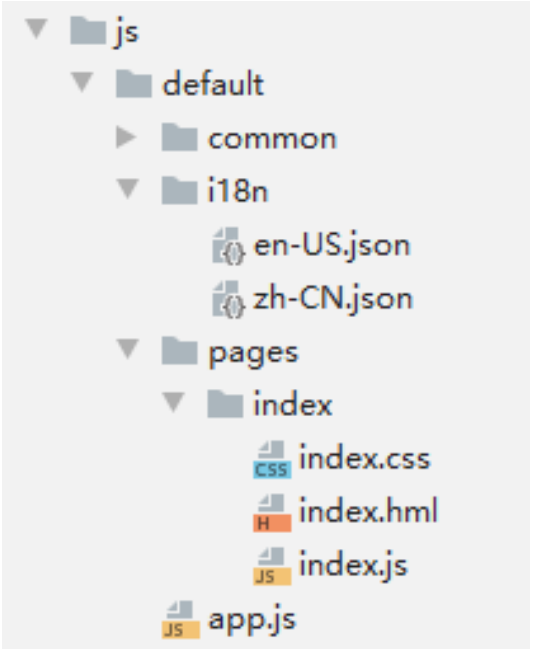
整体架构

- JS UI框架包括应用层（Application）、前端框架层（Framework）、引擎层（Engine）和平台适配层（Porting Layer）。



目录结构

- JS FA应用的JS模块(entry/src/main/js/module)的典型开发目录结构为：
  - app.js：用于全局JavaScript逻辑和应用生命周期管理；
  - pages：存放所有组件页面，每个页面由hml、css和js文件组成；
  - common：主要存放公共资源，如图片、视频、自定义组件等；
  - i18n：存放多语言的json文件，用于配置不同语言场景资源内容；
  - resources：存放资源配置文件，如全局样式、多分辨率加载等配置文件。




```
.
├── app.js
├── pages
│   ├── index
│   │   ├── index.hml
│   │   ├── index.css
│   │   └── index.js
│   └── detail(可选)
│       ├── detail.hml
│       ├── detail.css
│       └── detail.js
├── common(可选)
│   ├── xxx.png
│   ├── utils.js
│   ├── style.css
│   └── component
│       ├── componentA.hml
│       ├── componentA.css
│       └── componentA.js
├── resources(可选)
│   ├── styles
│   │   └── default.json
│   ├── res-ldpi.json
│   ├── res-xxhdpi.json
│   └── res-defaults.json
└── i18n(可选)
    ├── zh-CN.json
    └── en-US.json
```

JS FA基类

- AceAbility类是JS FA在HarmonyOS上运行环境的基类，继承自Ability。
- JS FA生命周期事件分为应用生命周期和页面生命周期，应用通过AceAbility类中setInstanceName()接口设置该Ability的实例资源，并通过AceAbility窗口进行显示以及全局应用生命周期管理。

```
public class MainAbility extends AceAbility {
    @Override
    public void onStart(Intent intent) {
        setInstanceName("JSComponentName");  // config.json配置文件中
ability.js.name的标签值。
        super.onStart(intent);
    }

    @Override
    public void onStop() {
        super.onStop();
    }
}
```

js标签配置

- js标签中包含了实例名称、页面路由和窗口样式等信息。name、pages和window等标签配置需在“config.json”配置文件中的“js”标签中完成设置。
- pages列表中第一个页面是应用的首页，即entry入口。
- 页面文件名不能使用组件名称，比如：text.hml、button.hml等。
- window用于定义与显示窗口相关的配置。

标签	类型	默认值	必填	描述
name	string	default	是	标识JS实例的名字。
pages	Array	-	是	路由信息
window	Object	-	否	窗口信息

js标签配置示例

```
{
  "app": {
    "bundleName": "com.huawei.player",
    "version": {
        "code": 1,
        "name": "1.0"
    },
    "vendor": "example"
  }
  "module": {
    ...
    "js": [
      {
        "name": "default",
        "pages": [
          "pages/index/index",
          "pages/detail/detail"
        ],
        "window": {
          "designWidth": 720,
          "autoDesignWidth": false
        }
      }
    ],
    "abilities": [
      {
        ...
      }
    ]
  }
}
```

应用资源访问规则

- 应用资源可通过绝对路径（“/”开头）或相对路径（“./”或“../“开头）的方式进行访问；

- 具体访问规则如下：
  - 引用代码文件，需使用相对路径，如：../common/utils.js；
  - 引用资源文件，推荐使用绝对路径。如：/common/xxx.png；
  - 公共代码文件和资源文件推荐放在common下，通过规则1和规则2进行访问；
  - CSS样式文件中通过url()函数创建<url>数据类型，如：url(/common/xxx.png)
  - 当代码文件A需要引用代码文件B时，如果代码文件A和文件B位于同一目录，则代码文件B引用资源文件时可使用相对路径，也可使用绝对路径；
  - 当代码文件A需要引用代码文件B时，如果代码文件A和文件B位于不同目录，则代码文件B引用资源文件时必须使用绝对路径。因为Webpack打包时，代码文件B的目录会发生变化。

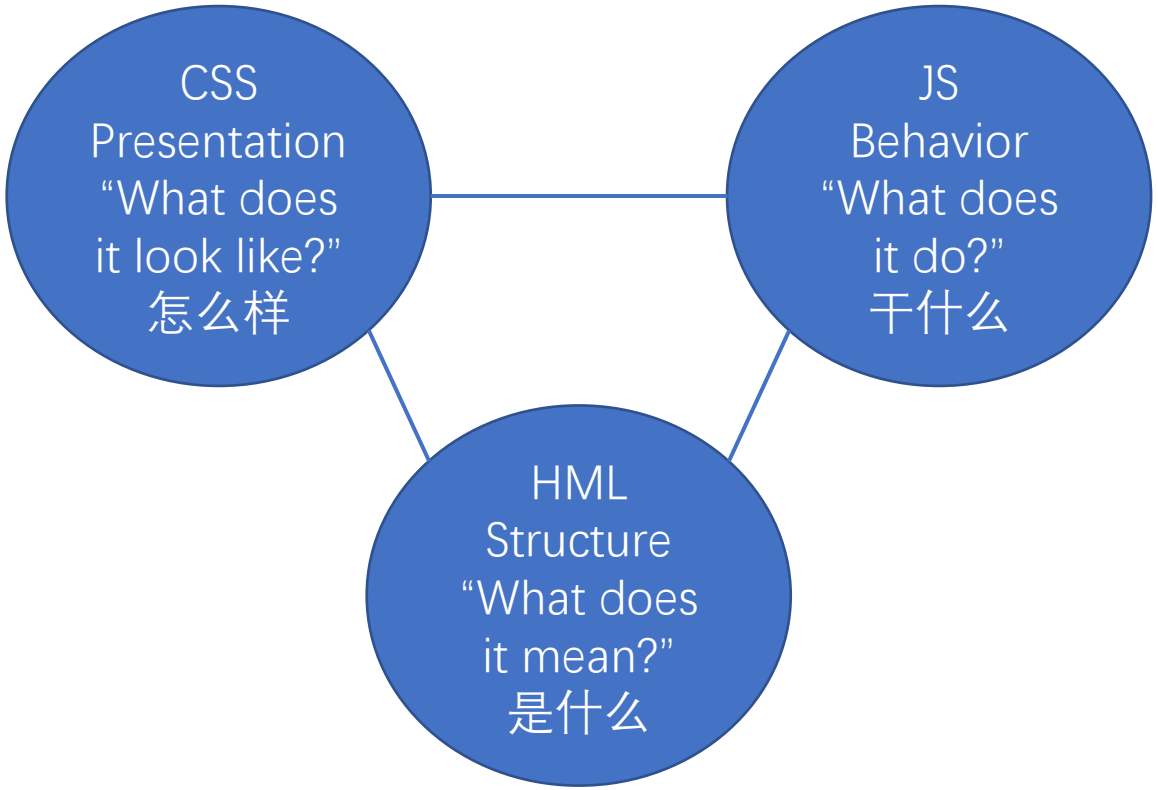
存储目录定义

- 应用使用文件存储接口访问文件时，可以通过使用特定scheme（只支持internal）来访问预定义的一些文件存取目录。对于不在下列目录下的文件访问将被拒绝（禁止使用../等方式访问父目录）。

目录类型	路径前缀	访问可见性	说明
临时目录	internal://cache/	仅本应用可见	可读写，随时可能清除，不保证持久性。一般用作下载临时目录或缓存目录。
应用私有目录	internal://app/	仅本应用可见	随应用卸载删除。
外部存储	internal://share/	所有应用可见	随应用卸载删除。其他应用在有相应权限的情况下可读写此目录下的文件。

目录结构

- 目录结构中文件分类如下：
  - .hml结尾的HML模板文件：用来描述当前页面的文件布局结构；
  - .css结尾的CSS样式文件：用于描述页面样式；
  - .js结尾的JS文件：用于处理页面和用户的交互。



HML语法参考

- HML（HarmonyOS Markup Language）是一套类HTML的标记语言，通过组件，事件构建出页面的内容。页面具备数据绑定、事件绑定、列表渲染、条件渲染和逻辑控制等高级能力。

- 页面结构如下：

```
<!-- xxx.hml -->
<div class="item-container">
  <text class="item-title">Image Show</text>
  <div class="item-content">
    <image src="/common/xxx.png" class="image"></image>
  </div>
</div>
```

HML语法参考

- 数据绑定：

```
<!-- xxx.hml -->
<text> {{content}} </text>
```

```
// xxx.js
export default {
  data: {
    content: 'Hello World!',
  },
}
```

HML语法参考

- 事件绑定：

```
<!-- xxx.hml -->
<div>
  <!-- 正常格式 -->
  <div onclick="clickfunc"></div>
  <!-- 缩写 -->
  <div @click="clickfunc"></div>
</div>
```

```
// xxx.js
export default {
  data: {
    obj: '',
  },
  clickfunc: function() {
    this.obj = 'Hello World';
  },
}
```

HML语法参考

- 列表渲染：for循环，默认$item代表数组中的元素, $idx代表数组中的元素索引；
- 条件渲染：if/elif/else和show，禁止在同一个元素上同时设置for和if属性；
- 逻辑控制块：block在构建时不会被当作真实的节点编译。block标签只支持for和if属性；
- 模板引用：HML可以通过element引用模板文件。
- ……

HML语法参考--组件介绍

- 组件名称对大小写不敏感，默认使用小写；
- 根据组件的功能，可以将组件分为以下四大类，用户也可以实现自定义组件：

组件类型	主要组件
基础组件	text、image、progress、rating、span、marquee、image-animator、divider、search、menu、chart
容器组件	div、list、list-item、stack、swiper、tabs、tab-bar、tab-content、list-item-group、refresh、dialog
媒体组件	video
画布组件	canvas

CSS语法参考

- CSS是描述HML页面结构的样式语言。所有组件均存在系统默认样式，也可在页面CSS样式文件中对组件、页面自定义不同的样式。
- 每个页面目录下存在一个与布局hml文件同名的css文件，用来描述该hml页面中组件的样式，决定组件应该如何显示。
- 为了模块化管理和代码复用，CSS样式文件支持 **@import** 语句，导入 CSS 文件。

CSS语法参考--声明样式

- 声明样式可以有内部样式或外部文件声明方式：
- 内部样式，支持使用style、class属性来控制组件的样式：

```
<!-- index.hml -->
<div class="container">
  <text style="color:red">Hello World</text>
</div>
```

- 外部文件声明样式，还可以引入合并样式文件：

```
/* style.css */
.title {
  font-size: 50px;
}
```

```
/* index.css */
@import '../../common/style.css';
.container {
  justify-content: center;
}
```

CSS语法参考--选择器

css选择器用于选择需要添加样式的元素，支持的选择器如下表所示：

选择器	样例	描述
类选择器 **.class**	.container	用于选择class="container"的组件。
Id选择器 **#id**	#titleId	用于选择id="titleId"的组件。
组件选择器 **tag**	text	用于选择text组件。
组合选择器 **,**	.title, .content	用于选择class="title"和class="content"的组件。
后代选择器 **#id .class tag**	#containerId .content text	非严格父子关系的后代选择器，选择具有id="containerId"作为祖先元素，class="content"作为次级祖先元素的所有text组件。如需使用严格的父子关系，可以使用“>”代替空格，如：#containerId>.content

各类选择器优先级由高到低顺序为：**内联样式 > id > class > tag**

CSS语法参考—伪类

css伪类是选择器中的关键字，用于指定要选择元素的特殊状态。如，**:disabled**状态可以用来设置元素的disabled属性变为true时的样式。

支持伪类组合，如，:focus:checked状态可以用来设置元素的focus属性和checked属性同时为true时的样式。

支持的单个伪类如下表所示，按照优先级降序排列：

名称	支持组件	描述
:disabled	支持disabled属性的组件	表示disabled属性变为true时的元素（不支持动画样式的设置）。
:focus	支持focusable属性的组件	表示获取focus时的元素（不支持动画样式的设置）。
:active	支持click事件的组件	表示被用户激活的元素，如：被用户按下的按钮、被激活的tab-bar页签（不支持动画样式的设置）。
:waiting	button	表示waiting属性为true的元素（不支持动画样式的设置）。
:checked	input[type="checkbox"、type="radio"]、 switch	表示checked属性为true的元素（不支持动画样式的设置）。

CSS语法参考—样式预编译

预编译提供了利用特有语法生成css的程序，可以提供变量、运算等功能，令开发者更便捷地定义组件样式，目前支持less、sass和scss的预编译。使用样式预编译时，需要将原css文件后缀改为less、sass或scss，如index.css改为index.less、index.sass或index.scss。

JS语法参考

- JS文件用来定义HML页面的业务逻辑，支持ECMA规范的JavaScrip语言(ES6语法) 。基于JavaScript语言的动态化能力，可以使应用更加富有表现力，具备更加灵活的设计。

JS语法参考—引入

- 模块声明，使用import方法引入功能模块：

```
import router from '@system.router';
```

- 代码引用，使用import方法导入js代码：

```
import utils from '../../common/utils.js';
```

JS语法参考—对象

应用对象

属性	类型	描述
$def	Object	使用this.$app.$def获取在app.js中暴露的对象。

页面对象

属性	类型	描述
data	Object/Function	页面的数据模型，类型是对象或者函数，如果类型是函数，返回值必须是对象。属性名不能以$或_开头，不要使用for, if, show, tid等保留字。data与private和public不能重合使用。
$refs	Object	持有注册过ref 属性的DOM元素或子组件实例的对象。
private	Object	页面的数据模型，private下的数据属性只能由当前页面修改。
public	Object	页面的数据模型，public下的数据属性的行为与data保持一致。
props	Array/Object	props用于组件之间的通信，可以通过<tag xxxx='value'>方式传递给组件；props名称必须用小写，不能以$或_开头，不要使用for, if, show, tid等保留字。目前props的数据类型不支持Function。
computed	Object	用于在读取或设置进行预先处理，计算属性的结果会被缓存。计算属性名不能以$或_开头，不要使用保留字。

JS语法参考—方法

■ 公共方法

属性	类型	参数	描述
$element	Function	id: string 组件id	获得指定id的组件对象，如果无指定id，则返回根组件对象。 用法： this.$element('xxx')：获得id为xxx的组件对象。 this.$element()：获得根组件对象。
$root	Function	无	获得顶级ViewModel实例。
$parent	Function	无	获得父级ViewModel实例。
$child	Function	id: string 组件id	获得指定id的子级自定义组件的ViewModel实例。 用法： this.$child('xxx')：获取id为xxx的子级自定义组件的ViewModel实例。

JS语法参考—方法

■ 数据方法

属性	类型	参数	描述
$set	Function	key: string value: any	添加新的数据属性或者修改已有数据属性。 用法：this.$set('key',value)：添加数据属性。
$delete	Function	key: string	删除数据属性。 用法：this.$delete('key')：删除数据属性。

■ 事件方法

属性	类型	参数	描述
$watch	Function	data: string callback: 函数名，回调函数里有两个参数，第一个参数为属性新值，第二个参数为属性旧值	观察data中的属性变化，如果属性值改变，触发绑定的事件。 用法： this.$watch('key', callback)

JS语法参考—示例代码

通过$refs获取DOM元素

```
<!-- index.hml -->
<div class="container">
  <image-animator class="image-player" ref="animator" images="{{images}}" duration="1s" onclick="handleClick"></image-animator>
</div>
```

```
// index.js
export default {
  data: {
    images: [
      { src: '/common/frame1.png' },
      { src: '/common/frame2.png' },
      { src: '/common/frame3.png' },
    ],
  },
  handleClick() {
    const animator = this.$element('animator'); //获取id属性为animator 的DOM元素
    const state = animator.getState();
    if (state === 'paused') {
      animator.resume();
    } else if (state === 'stopped') {
      animator.start();
    } else {
      animator.pause();
    }
  },
};
```

JS语法参考—生命周期接口

■ 页面生命周期

属性	类型	参数	返回值	描述	触发时机
onInit	Function	无	无	页面初始化	页面数据初始化完成时触发，只触发一次。
onReady	Function	无	无	页面创建完成	页面创建完成时触发，只触发一次。
onShow	Function	无	无	页面显示	页面显示时触发。
onHide	Function	无	无	页面隐藏	页面消失时触发。
onDestroy	Function	无	无	页面销毁	页面销毁时触发。
onBackPress	Function	无	Boolean	返回按钮动作	当用户点击返回按钮时触发。 返回true表示页面自己处理返回逻辑。 返回false表示使用默认的返回逻辑。 不返回值会作为false处理。

JS语法参考—生命周期接口

- 应用生命周期

属性	类型	参数	返回值	描述	触发时机
onCreate	Function	无	无	应用创建	当应用创建时调用。
onDestroy	Function	无	无	应用退出	当应用退出时触发。

多语言支持

- 基于开发框架的应用会覆盖多个国家和地区，开发框架支持多语言能力后，可以让应用开发者无需开发多个不同语言的版本，就可以同时支持多种语言的切换，为项目维护带来便利。
- 开发者仅需要通过定义资源文件和引用资源两个步骤，就可以使用开发框架的多语言能力。
- 资源文件命名为“语言-地区.json”格式，开发框架无法在应用中找到系统语言的资源文件时，默认使用en-US.json中的资源内容。
- 在应用开发的页面中使用多语言的语法，包含简单格式化和单复数格式化两种，都可以在hml或js中使用。

根据设备分辨率加载图片

- 基于开发框架的应用可能会应用于多种不同的设备，由于不同的设备有不同的DPI，应用在不同DPI的设备上可能需要配置不同的图片资源。
- 开发框架支持基于不同DPI的设备加载不同图片资源，开发者仅需要通过定义资源文件和引用资源两个步骤，即可以使用开发框架的基于设备DPI加载不同图片资源的功能。
- 在/resources目录下定义不同DPI设备对应的资源文件，开发框架采用json文件保存资源定义：例如适用于低密度（ldpi）屏幕（~120dpi）的资源文件为res-ldpi.json，适用于超超高密度（xxhdpi）屏幕（~480dpi）的资源文件为res-xxhdpi.json。
- 以160dpi为基准密度，低密度为0.75*基准密度。
- 如果当前设备的DPI不完全匹配表1中定义的DPI，那么将选取更接近当前设备DPI的资源文件。
- 在应用开发的hml和js文件中使用$r的语法，可以对图片资源进行格式化，针对不同DPI的设备获取不同的图片资源。

根据设备分辨率加载图片

不同像素密度的配置限定符

密度限定符	说明
ldpi	表示低密度屏幕（~120dpi）（0.75*基准密度）
mdpi	表示中密度屏幕（~160dpi）（基准密度）
hdpi	表示高密度屏幕（~240dpi）（1.5*基准密度）
xhdpi	表示加高密度屏幕（~320dpi）（2.0*基准密度）
xxhdpi	表示超超高密度屏幕（~480dpi）（3.0*基准密度）
xxxhdpi	表示超超超高密度屏幕（~640dpi）（4.0*基准密度）

内容小结

- HML语法及常用组件
- CSS语法及选择器
- JS语法及常用事件

练习巩固

问题1：以下哪项不属于组件通用事件？（ ）

A. touchstart

B. touchmove

C. dbclick

D. longpress

问题2：pages标签配置定义每个页面的路由信息，每个页面由页面路径和页面名组成，页面的文件名就是页面名。

A. 正确

B. 错误

思考挑战

创建两个页面A、B，实现页面间的跳转，并观察生命周期变化。

THANKS

更多学习视频，关注宅客学院……