



UWP应用的可适配性

北京理工大学计算机学院
金旭亮

UWP如何实现可适配性?

“**可适配性 (Adaptive)**”，指UWP应用具有这样的一种技术特性——可以不加修改地跑在不同设备上，并且这些设备可以跑不同的Windows 10版本。

版本
适配性

界面
适配性

平台
适配性

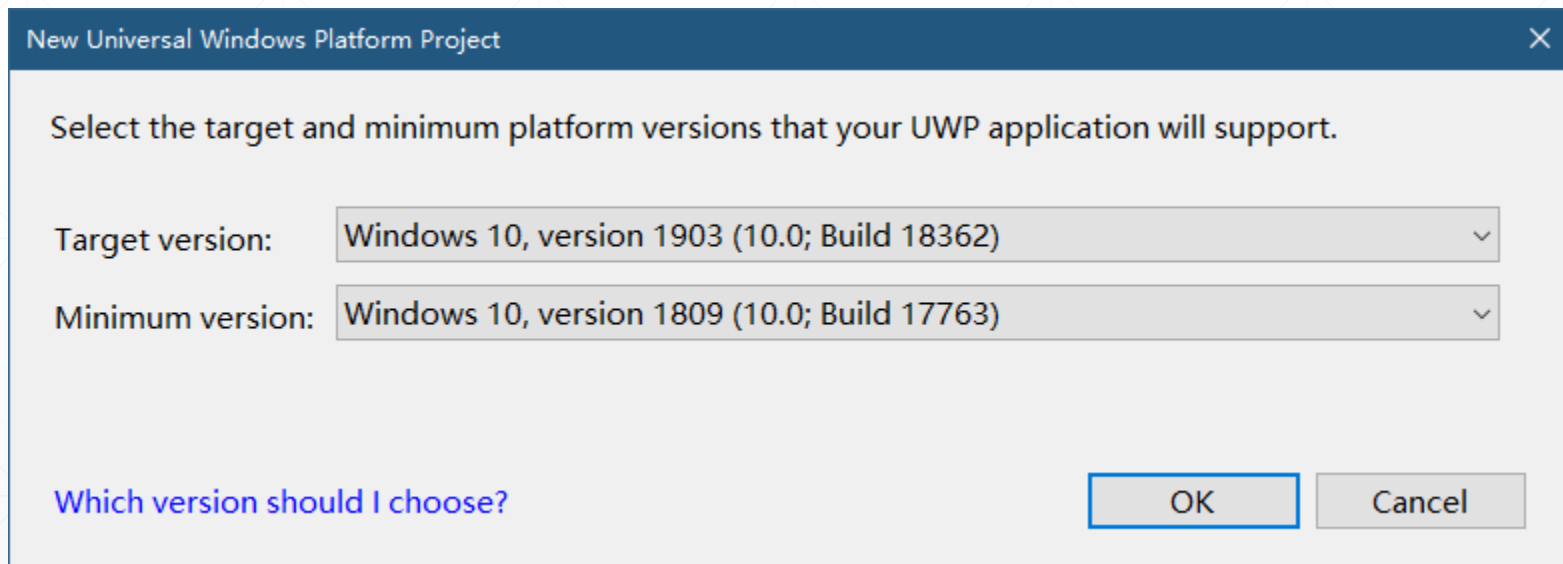


实现UWP应用的版本适配性

UWP如何应对Windows 10的版本差异?

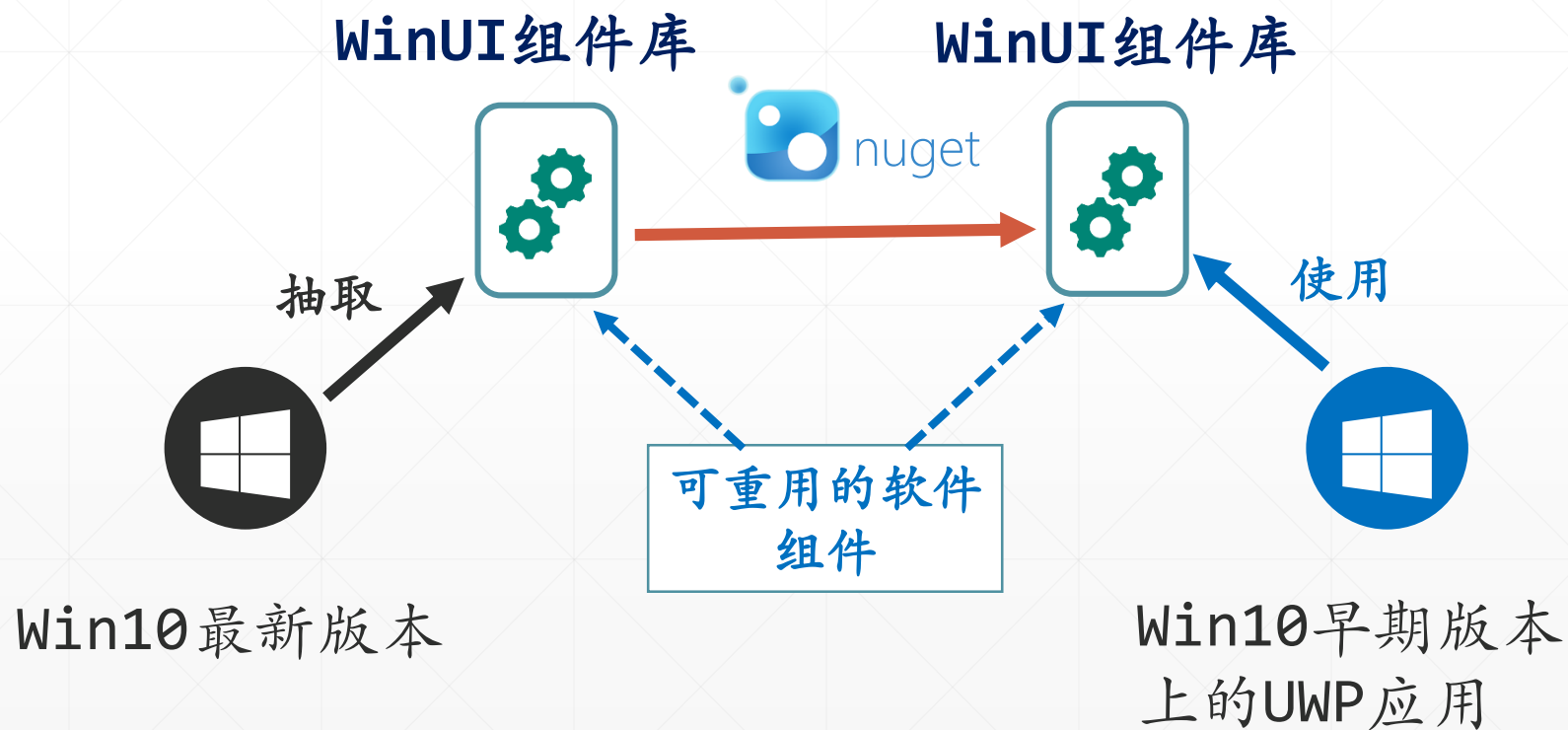
解决方案一：显式指定一个可用的版本范围

通过指定“目标版本（Target version，即最高版本）”与最低版本，限定UWP应用可使用的API集合。



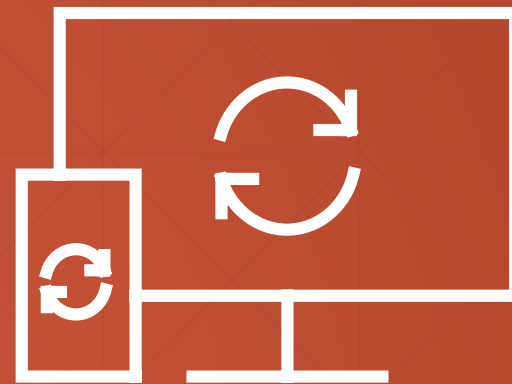
解决方案二：发布外部兼容组件包

通过创建独立于Windows 10的外部软件组件，实现版本兼容性。



Android Support Library

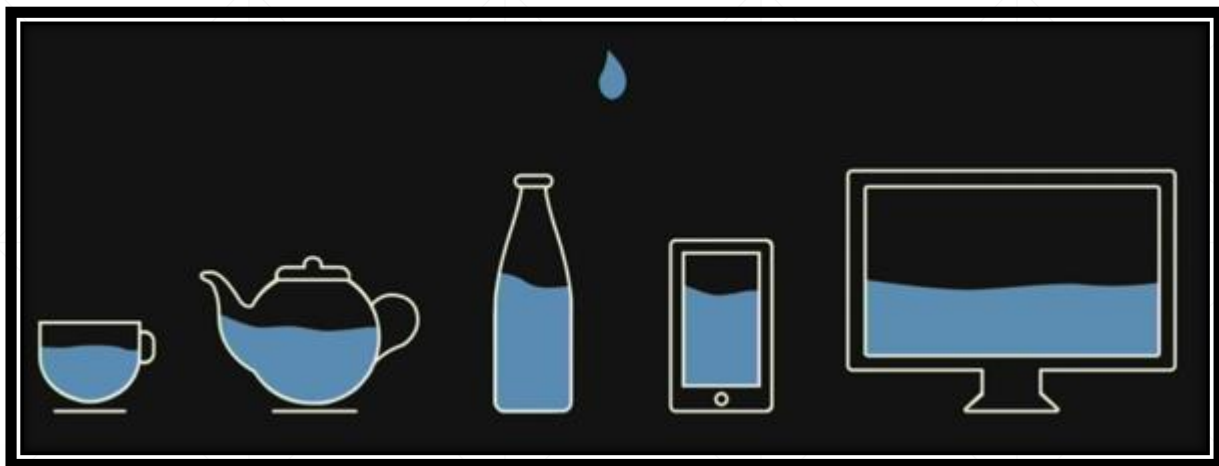
UWP采用的技术方案，与Android采用的“兼容包”方案“一模一样”。



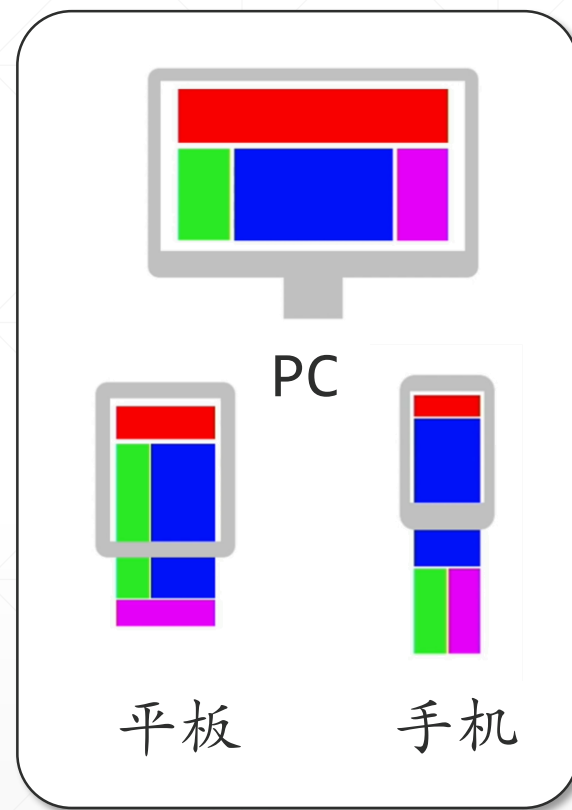
实现UWP界面的可适配性

面对着大大小小的显示屏幕，如何应对？

实现界面适配性的关键——响应式设计



你把水装进**杯子**，它就呈现**杯子**的形状
你把水装进**茶壶**，它就呈现**茶壶**的形状
你把水装进**瓶子**，它就呈现**瓶子**的形状
你把水装进**(.....)**，它就呈现**(.....)**的形状



同一应用，在不同的设备使用不同的布局

构建响应式的UWP应用

Fluent Design System

指导、设计
和开发

微软公司
开源社区

UWP官方和社区控件



支持响应式
特性



UWP应用使用者

使用



UWP应用

基于

开发



UWP应用开发者



实现UWP应用的平台适配性

各种各样的设备，拥有不一样的技术特性，怎样处理？

什么叫平台适配性？

一个具有平台适配性的应用，能够依据它所运行的设备种类，具有不同的功能和技术特性。



在普通PC上使用键盘+鼠标
输入



在Surface上则可使用手指
触摸和手写笔输入

“contract”的概念

UWP应用可调用的所有API，被划分为逻辑上紧密相关的“区域”，这些区域被称为“**contracts**”。

因此，“API contract”，其实就是一组实现了某个特定技术特性或功能的相关API的集合。

所有设备都可以使用这个Contract中的API，约占全部UWP API的85%。

Universal

XboxLive
Storage

Wallet

Dual SIM Tile

Scanner
Device

...

我们可以编写代码，在UWP应用运行的时候，动态地检查是否可以使用特定Contracts中的API。

```
0 references
protected override void OnNavigatedTo(NavigationEventArgs e)
{
    base.OnNavigatedTo(e);

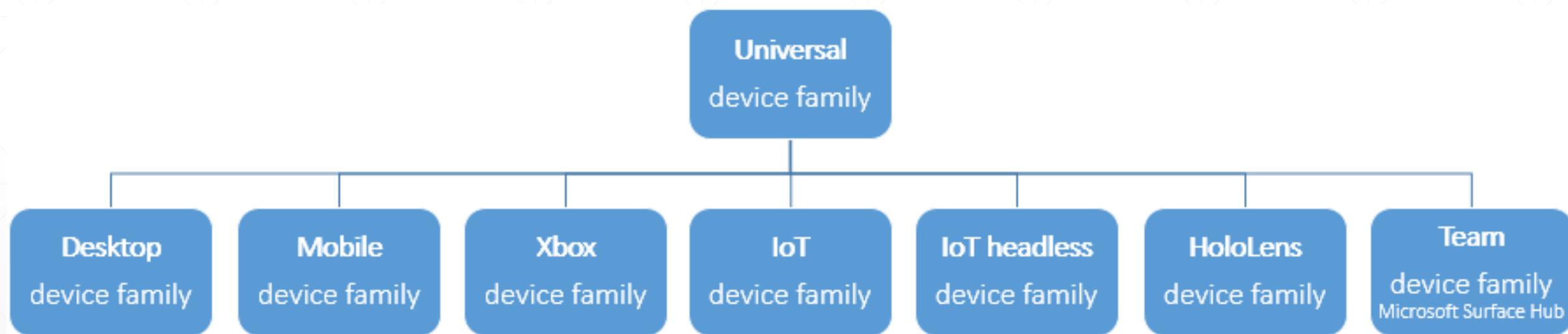
    var allContracts = APIContractsFactory.GetAPIContracts();

    foreach(var contract in allContracts)
    {
        if(ApiInformation.IsApiContractPresent(contract, 1))
        {
            availableContracts.Add(contract);
        }
    }
}
```

“可适配性代码 (Adaptive Code)”
的典型示例

UWP的设备家族

UWP支持多种设备，它使用一个专门的术语——“device family（设备家族）”来对这些设备进行分类，所有API都会标明它适用于哪个设备家族。



<https://docs.microsoft.com/en-us/uwp/extension-sdks/device-families-overview>

UWP的“设备家族标识字符串”

Device family	Example	Identifier string
Universal	N/A	Windows.Universal
Desktop	Surface Studio	Windows.Desktop
Mobile	Lumia 950	Windows.Mobile
Xbox	Xbox One X	Windows.Xbox
Holographic	HoloLens	Windows.Holographic
IoT	Raspberry Pi 3	Windows.IoT
IoT Headless	Minnowboard Max	Windows.IoTHeadless
Team	Surface Hub	Windows.Team

每个设备家族，对应唯一的一个字符串

UWP应用需要指定针对的设备家族

1

Package.appxmanifest

```
<Dependencies>  
  <TargetDeviceFamily Name="Windows.Universal" ... />  
</Dependencies>
```



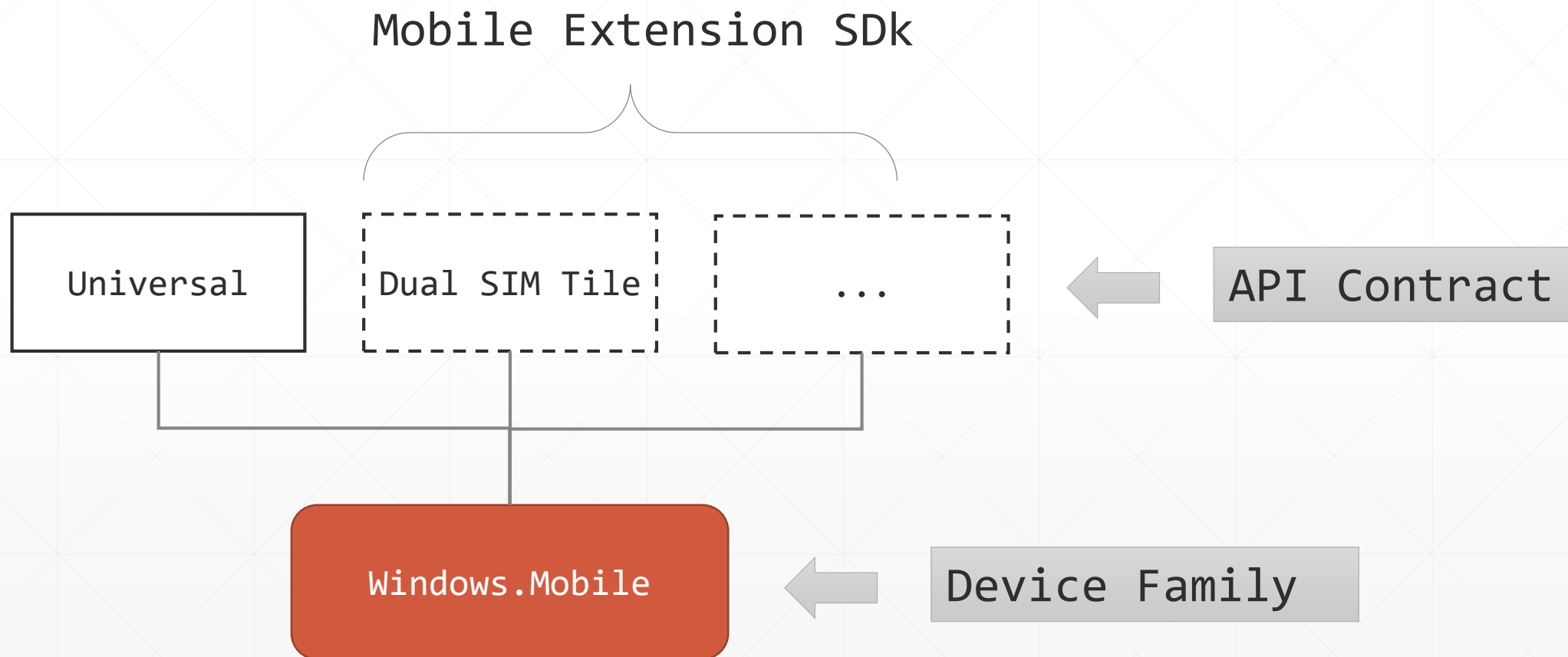
2

Package.appxmanifest

```
<Dependencies>  
  <TargetDeviceFamily Name="Windows.Holographic" ... />  
  <TargetDeviceFamily Name="Windows.Desktop"... />  
</Dependencies>
```



Contract与Device Family如何相互配合?



UWP API文档中的类型

ImageScanner Class


Namespace: [Windows.Devices.Scanners](#)

Assemblies: [Windows.Devices.Scanners.dll](#), [Windows.dll](#)

Represents the properties of images to scan.

 Edit

C#

 Copy

```
public sealed class ImageScanner
```

Attributes [ContractVersionAttribute](#), [DualApiPartitionAttribute](#),
[MarshalingBehaviorAttribute](#), [StaticAttribute](#), [ThreadingAttribute](#)

Windows 10 requirements

Device family	Windows Desktop Extension SDK (introduced v10.0.10240.0)
API contract	Windows.Devices.Scanners.ScannerDeviceContract (introduced v1)

每一个类型，都会标出它所属的设备家族和API Contract

基于可适配性标准给UWP应用分类

