Работа с изображениями. Элемент Image

Последнее обновление: 26.03.2017

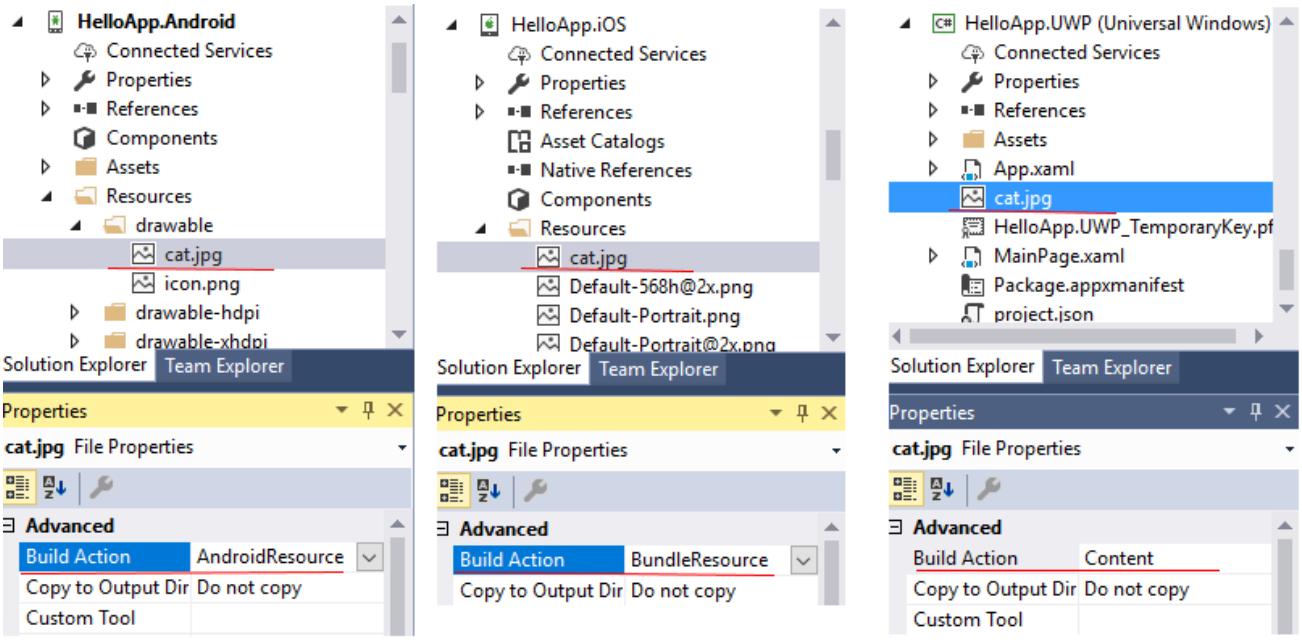
Для вывода изображение имеется элемент **Image**. Но перед выводом изображения его надо добавить в проекты, причем в проект для каждой отдельной ос.

Локальные изображения

Возьмем какое-нибудь изображение. В проекте для Androida добавим файл изображения в папку *Resources/Drawable*. А в окне Properties установим у этого изображения **Build Action: AndroidResource**.

Также добавим изображение в проект для iOS в папку *Resources* и установим у него свойство **Build Action: BundleResource**

Для проекта для UWP изображение просто добавляется в корневую папку проекта, а в окне свойств устанавливаем для изображения **Build Action: Content**. В итоге получится следующее:



А простейший код будет выглядеть так:

```
class MainPage : ContentPage
{
    public MainPage()
    {
        Image image = new Image { Source = "cat.jpg" };
        this.Content = image;
    }
}
```

Аналог в xaml:

```
<?xml version="1.0" encoding="utf-8" ?>
<ContentPage xmlns="http://xamarin.com/schemas/2014/forms"
```

```
            xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
            x:Class="HelloApp.MainPage">
  <Image Source="cat.jpg" />
</ContentPage>
```

В то же время такой подход может быть неудобен. Например, не очень было бы хорошо захломлять проект для UWP файлами и изображений, и вполне вероятно что мы бы предпочли создать в проекте специальную папку, к примеру, Images. А в нее уже добавлять файлы изображений. То же самое может относиться и к проекту для iOS. Однако в этом случае у нас была бы рассинхронизация в путях к файлу изображений. Так, в Android путь был бы по прежнему *cat.jpg*, а в для проекта для UWP - *images/cat.jpg*.

И если пути к изображению отличаются в засимости от операционной системы, то мы могли бы установить путь с помощью метода `Device.OnPlatform`:

```
image.Source =  Device.OnPlatform(
        iOS: ImageSource.FromFile("Images/cat.jpg"),
        Android:  ImageSource.FromFile("cat.jpg"),
        WinPhone: ImageSource.FromFile("Images/cat.png"));
```

Размеры изображений

При использовании изображений надо учитывать размеры устройств. Например, для смартфона было бы не очень оптимально использовать изображение шириной 1000 пикселей. Возможно, такое изобржение следует масштабировать, чтобы оно блее оптимально выглядело на данном типе устройств.

На разных платформах есть свои принципы создания изображений для устройств с разной шириной экрана.

iOS

iOS применяет систему наименовний, учитывая суффикс в имени файла. Операционная система выбирает определенный файл изображения по имени на основании разрешения экрана:

- Изображения с именами без суффикса выбираются для устройств с 160 DPI (точек на дюйм)
- С суффиксом **@2x** выбираются для устройств с 320 DPI
- С суффиксом **@3x** выбираются для устройств с 480 DPI

Например, в нашем случае при имени файла *wild.jpg* нам фактически надо создать три версии этого файла:

- wild.jpg — размером 160 пикселей в ширину и высоту
- wild@2x.jpg — размером 320 пикселей в ширину и высоту
- wild@3x.jpg — размером 480 пикселей в ширину и высоту

Если же iOS не найдет версию файла с нужным суффиксом, то система берет ту версию файла, которая имеется, и должным образом ее масштабирует.

Android

Android применяет другой подход. Здесь файл надо положить в одну из подпапок в каталоге **Resources**:

- **drawable-hdpi**: для устройств с расширением 240 DPI
- **drawable-xhdpi**: для устройств с расширением 320 DPI
- **drawable-xxhdpi**: для устройств с расширением 480 DPI
- **drawable-xxxhdpi**: для устройств с расширением 640 DPI

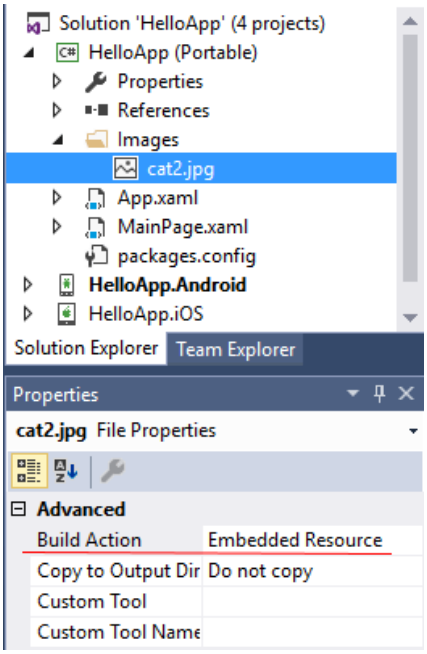
Поэтому в идеале нам надо определить 4 версии файла wild.jpg и положить их в соответствующую папку:

- drawable-hdpi/wild.jpg - шириной и высотой в 240 пикселей
- drawable-xhdpi/wild.jpg - шириной и высотой в 320 пикселей
- drawable-xxhdpi/wild.jpg - шириной и высотой в 480 пикселей
- drawable-xxxhdpi/wild.jpg - шириной и высотой в 640 пикселей

Embedded images

Встроенные изображения (embedded images) в отличие от выше рассмотренных локальных изображений добавляются непосредственно в разделяемую сборку в качестве ресурса. То есть нам надо добавить файл только в один проект - Portable.

Так, создадим в главном проекте новый каталог Images и добавим в него файл изображения:



После добавления изображения в панели свойств для поля **Build Action** установим значение **Embedded Resources**.

Выведем изображение в коде C#:

```
public partial class MainPage : ContentPage
{
    public MainPage()
    {
        Image image = new Image();
        image.Source = ImageSource.FromResource("HelloApp.Images.cat2.jpg");
        Content = image;
    }
}
```

Для получения ресурса изображения применяется метод `ImageSource.FromResource()`. Обратите внимание на путь, который в него передается. Этот путь начинается с названия проекта, то есть HelloApp. Дальше идет путь к изображению внутри проекта. Название проекта и название папок в этом пути отделяются точками.

Для XAML по умолчанию подобная возможность отсутствует, и нам надо вручную писать специальное расширение для XAML, которое позволит транслировать строковый путь в нужный нам объект.

Для этого добавим в проект следующий класс:

```
using System;
using Xamarin.Forms;
using Xamarin.Forms.Xaml;

namespace HelloApp
{
    [ContentProperty("Source")]
    public class ImageResourceExtension : IMarkupExtension
    {
        public string Source { get; set; }

        public object ProvideValue(IServiceProvider serviceProvider)
        {
            if (Source == null)
            {
                return null;
            }
            var imageSource = ImageSource.FromResource(Source);

            return imageSource;
        }
    }
}
```

Применим класс для загрузки изображения в коде xaml:

```
<?xml version="1.0" encoding="utf-8" ?>
<ContentPage xmlns="http://xamarin.com/schemas/2014/forms"
             xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
             xmlns:local="clr-namespace:HelloApp;assembly=HelloApp"
             x:Class="HelloApp.MainPage">
    <Image Source="{local:ImageResource HelloApp.Images.cat2.jpg}" />
</ContentPage>
```

Загрузка из сети

Кроме локальных картинок Xamarin также поддерживает загрузку из сети:

```
Image image = new Image();
image.Source = new UriImageSource
{
    CachingEnabled = false,
    Uri = new System.Uri("http://www.someserver/someimage.png")
};
```

Кроме url изображения также можно задать параметры кэширования. Выражение `CachingEnabled = false` отключает кэширование. Но мы также можем включить его и установить период кэширования:

```
Image image = new Image();
image.Source = new UriImageSource
{
    CachingEnabled = true,
    CacheValidity = new System.TimeSpan(2,0,0,0),
    Uri = new System.Uri("http://www.someserver/someimage.png")
};
```

Параметр `CacheValidity` указывает, сколько будет действовать кэширование - в данном случае 2 дня. Без установки параметра по умолчанию кэширование длится 24 часа.

Свойство Aspect

Свойство **Aspect** позволяет задать принцип масштабирования изображения при его выводе на экран. Это свойство в качестве значения принимает одну из констант из одноименного перечисления **Aspect**:

- `AspectFit`: значение по умолчанию. Если стандартное изображение не вписывается в экран (например, его ширина больше ширины экрана), то оно масштабируется с сохранением аспектного отношения (отношение ширины к длине)
- `Fill`: растягивает изображение по ширине или длине без сохрнения аспектного отношения
- `AspectFill`: сохраняет аспектное отношение, но вырезает из него ту часть, которая вписывается в экран

Выше на рисунке демонстрировалось значение по умолчанию - AspectFit. Теперь попробуем AspectFill:

```
<Image Source="wild.jpg" Aspect="AspectFill" />
```

В итоге результат будет несколько отличаться:

Назад **Содержание** **Вперед**

Так вот как выглядит Кот Админа!!)))

• Ответить • Поделиться ›

Erazer • 5 месяцев назад

Добрый день. При создании нескольких окон в каждое вставил по картинки весом 300+ кб. Переход на другую страницу осуществляется крайне тяжко и анимация перехода идет с задержкой, а при добавлении еще одного окна выходит исключение. Как я понял проблема в весе изображения. Обрезал все картинки до 100 кб и переход осуществляется быстро, а анимация без задержек, так же нет больше исключения! В сборки стоит ограничение на вес или что то еще? как можно увеличить размер приложения? Почитал англоязычные форумы там пишут что нужно место на куче увеличить!

• Ответить • Поделиться ›

Erazer • 6 месяцев назад

Добрый день, Подскажите а как к примеру установить фон и поверх активную кнопку. Я видимо много чего не понимаю и разбираюсь путем экспериментов. я пытаюсь одновременно создать и фон и на него наложить кнопку но выводит что то одно.

```
x:Class="HelloApp.MainPage">
<stacklayout>
<image source="StartFon"/>
<button text="Start" fontsize="Large" borderwidth="1" horizontaloptions="Center" verticaloptions="CenterAndExpand"/>
</stacklayout>
</contentpage>
```

• Ответить • Поделиться ›

Metanit Модератор ➚ Erazer • 6 месяцев назад

по-моему, надо использовать какой-нибудь другой контейнер компоновки, например, RelativeLayout, который позволяет
это сделать

• Ответить • Поделиться ›

Женя Быканов • 6 месяцев назад

Привет, в чем может быть проблема? На андроиде 4.0.1 изображения все выводит нормально, но на андроиде 5.0.1 часть изображений из списка не отображается.

• Ответить • Поделиться ›

Viktor Bylbas • 7 месяцев назад

Подскажите, как можно получить размер изображения

```
var map = new Image()
{
Source = ImageSource.FromResource(path)
};
var h = map.Height; // Возвращает -1
```

• Ответить • Поделиться ›

Serega Maleev • 9 месяцев назад

Подскажите как правильно присвоить изображение кнопке. Есть button я хочу присвоить ему изображение через свойство image но никак не выходит..

• Ответить • Поделиться ›

Mikhail Ukhin ➚ Serega Maleev • 9 месяцев назад

<button absolutelayout.layoutbounds=".5,.16,1,.1" absolutelayout.layoutflags="All" image="register.png" clicked="Button_Clicked"/>
у меня работает так без проблем

• Ответить • Поделиться ›

Serega Maleev ➚ Mikhail Ukhin • 9 месяцев назад

У меня как оказалось позже .PNG не отображается, с .Jpg все нормально

• Ответить • Поделиться ›

Иван Гришаев • год назад

Что-то у меня не отображается изображение. И в xaml его и кодом, ноль рекции, спокойно компилит и не выводит. Хотя там всякие блоки, кнопки, все гуд.
<image source="1.jpg" grid.column="0" grid.row="0" grid.columnspan="1" aspect="AspectFill"/>
Вроде и картинку разместил в ресурсах, там сразу по умолчанию стоит Build Action: AndroidResource. Чего не так то?

^ | v • Ответить • Поделиться ›

Mikhail ➚ Иван Гришаев • год назад

Имел ту же проблему. Создайте в общем проекте папку Resources и положите файл еще туда. Не используйте дефис для файлов в drawable-папке андроидовского проекта.

^ | v • Ответить • Поделиться ›

Makarkin & Partners / Макаркин • год назад

а с кэшэм для XAML есть пример?

^ | v • Ответить • Поделиться ›

Anton ➚ Makarkin & Partners / Макаркин • год назад

Присоединяюсь к вопросу.

^ | v • Ответить • Поделиться ›

Alex ➚ Anton • год назад

похоже общего кода для всех платформ нет. тут придется для каждой платформы специфический код писать в каждом проекте, ну или IF _PLATFORM_ писать для каждой поддерживаемой платформы :(

^ | v • Ответить • Поделиться ›

Erazer • 6 месяцев назад

Привет, подскажите как устанавливать размеры для изображения? Свойство аспект вообще не помогает...

^ | v • Ответить • Поделиться ›

Metanit Модератор ➚ Erazer • 6 месяцев назад

Попробуйте использовать свойства WidthRequest и HeightRequest

^ | v • Ответить • Поделиться ›