



Конвертеры значений

Последнее обновление: 16.06.2016



JOIN IN

Привязка позволяет связать два свойства, которые совпадают по типу данных. Например, свойство Text у класса Entry и свойство Text класса Label представляют тип string. Поэтому тут привязка проходит без проблем. Но в других случаях тип связанных свойств может отличаться. В большинстве случаев инфраструктура Xamarin сама знает, как конвертировать данные простейших типов. Однако в каких-то ситуациях этого оказывается недостаточно, особенно когда необходимо применить к значению привязки дополнительное форматирование.

Например, пусть у нас элемент Label привязан к элементу DatePicker и выводит выбранную дату. По умолчанию дата выводится в формате "MM/dd/yyyy hh:mm:ss". Это не очень удобный формат. Возможно, мы захотим использовать совсем другие форматы.

Для преобразования одного значения к другому в процессе привязки используются специальные классы - конвертеры значений. К примеру, добавим в наш проект новый класс конвертера значений:

```
using System;
using System.Globalization;
using Xamarin.Forms;

namespace HelloApp
{
    public class DateTimeToLocalDateConverter : IValueConverter
    {
        public object Convert(object value, Type targetType, object parameter, CultureInfo culture)
        {
            return ((DateTime)value).ToString("dd.MM.yyyy");
        }
        public object ConvertBack(object value, Type targetType, object parameter, CultureInfo culture)
        {
            return DateTime.Now.ToString("dd.MM.yyyy");
        }
    }
}
```

Прежде всего класс конвертера значений должен реализовать интерфейс **IValueConverter**. Этот интерфейс определяет два метода: `Convert()`, который преобразует пришедшее от привязки значение в тот тип, который понимается приемником привязки, и `ConvertBack()`, который выполняет противоположную операцию.

Оба метода принимают четыре параметра:

- `object value`: значение, которое надо преобразовать
- `Type targetType`: тип, к которому надо преобразовать значение `value`
- `object parameter`: вспомогательный параметр
- `CultureInfo culture`: текущая культура приложения

В примере выше метод `Convert` возвращает дату в виде строки в формате "dd.MM.yyyy". То есть мы ожидаем, что в качестве параметра `value` будет передаваться объект `DateTime`.

Метод `ConvertBack` в данном случае не имеет значения, и поэтому он просто возвращает строковое представление текущей даты.

Теперь применим этот конвертер 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"
  xmlns:local="clr-namespace:HelloApp;assembly=HelloApp"
  x:Class="HelloApp.MainPage">
  <ContentPage.Resources>
    <ResourceDictionary>
      <local:DateTimeToLocalDateConverter x:Key="dateConverter" />
    </ResourceDictionary>
  </ContentPage.Resources>
  <StackLayout>
    <Label Text="{Binding Source={x:Reference Name=datePicker}, Path=Date}" />
    <Label Text="{Binding Source={x:Reference Name=datePicker},
      Path=Date,
      Converter={StaticResource dateConverter}}" />
    <DatePicker x:Name="datePicker" Format="D" />
  </StackLayout>
</ContentPage>
```

Так как класс конвертера располагается в пространстве имен текущего проекта, то, чтобы класс был доступен в xaml, надо подключить текущее пространство имен.

```
xmlns:local="clr-namespace:HelloApp;assembly=HelloApp"
```

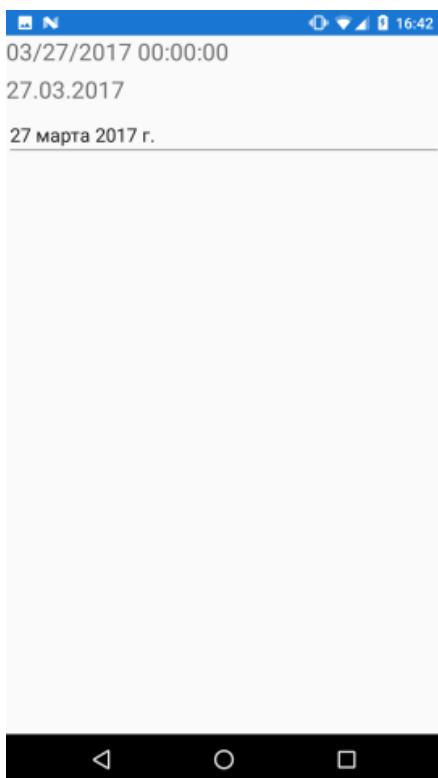
В моем случае пространством имен является HelloApp, и оно проецируется на суффикс local

Затем в ресурсах создается объект привязки с ключом dateConverter.

И далее в самом выражении привязки мы подключим данный ресурс в качестве конвертера:

```
Converter={StaticResource dateConverter}
```

При этом здесь привязка идет к элементу выбора даты для двух объектов Label. Но только второй объект применяет конвертер:



Возьмем другой случай. Допустим, нам надо привязать цвет метки к вводимому в текстовое поле значение. Для этого определим следующий конвертер:

```
using System;
using System.Globalization;
using Xamarin.Forms;

namespace HelloApp
{
    public class StringToColorConverter : IValueConverter
    {
        public object Convert(object value, Type targetType, object parameter, CultureInfo culture)
        {

```

```

Color color = Color.White;
string colorStr = value.ToString().ToLower();
switch(colorStr)
{
    case "красный":
        color = Color.Red;
        break;
    case "желтый":
        color = Color.Yellow;
        break;
    case "зеленый":
        color = Color.Green;
        break;
    case "синий":
        color = Color.Blue;
        break;
}
return color;
}
public object ConvertBack(object value, Type targetType, object parameter, CultureInfo culture)
{
    return Color.White;
}
}
}

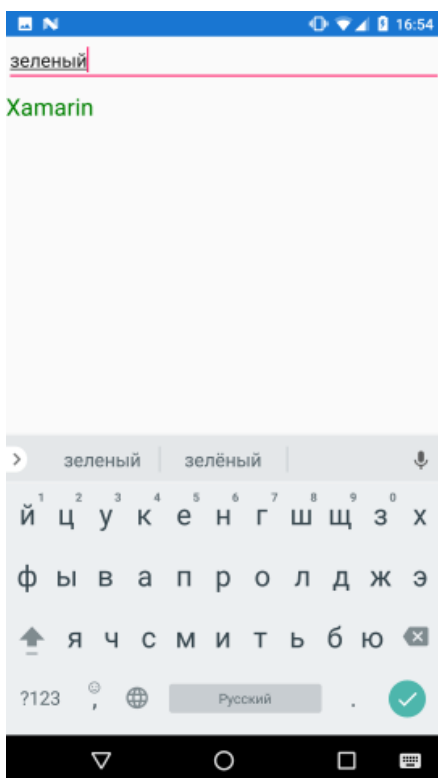
```

Если вводится определенная строка, то конвертер возвращает определенный цвет. Теперь применим этот конвертер 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"
    xmlns:local="clr-namespace:HelloApp;assembly=HelloApp"
    x:Class="HelloApp.MainPage">
    <ContentPage.Resources>
        <ResourceDictionary>
            <local:StringToColorConverter x:Key="colorConverter" />
        </ResourceDictionary>
    </ContentPage.Resources>
    <StackLayout>
        <Entry x:Name="entry1" Text="Red" />
        <Label x:Name="label1" Text="Xamarin"
            TextColor="{Binding Source={x:Reference Name=entry1},
                Converter={StaticResource colorConverter},
                Path=Text}" />
    </StackLayout>
</ContentPage>

```





10%

JOIN IN



[Назад](#) [Содержание](#) [Вперед](#)



Кабель ВВГНГ-LS!

КУРС iOS РАЗРАБОТЧИКА

Высокая востребованность на рынке IT!

ОБРАЗОВАТЕЛЬНЫЙ ЦЕНТР ПВТ

Яндекс.Директ

4 Комментариев metanit.com

1 Войти ▾

♥ Рекомендовать  Поделиться

Лучшее в начале ▾



Присоединиться к обсуждению...

ВОЙТИ С ПОМОЩЬЮ

ИЛИ ЧЕРЕЗ DISQUS ?

**Алексей Павлов** • месяц назад

По примеру пытаюсь привязать конвертер к свойству IsVisible у StackLayout. При отладке попадет в метод Convert, все ок, но параметр value почему то равен null. В качестве value пытаюсь передать значение свойства Text у Label. Вот неполный код XAML.

```
<label x:name="Status" text="{Binding ClaimStatus}"></label>
<stacklayout orientation="Horizontal" isvisible="{Binding Source={x:Reference Status}, Converter={StaticResource statusConverter}, Path=Text}">
<button backgroundColor="#D50000" horizontaloptions="StartAndExpand" bindingcontext="{Binding Source={x:Reference claimsList}, Path=BindingContext}" command="{Binding CloseClaimCommand}" commandparameter="{Binding Source={x:Reference Id}, Path=Text}" textcolor="White" heightrequest="35" text="Заккрыть">
</button>
</stacklayout>
```

^ | ▾ • Ответить • Поделиться ›

**Алексей Павлов** → Алексей Павлов • месяц назад

В StackLayout в место isvisible="{Binding Source={x:Reference Status}, Converter={StaticResource statusConverter}, Path=Text}" добавил IsVisible="{Binding ClaimStatus, Converter={StaticResource statusConverter}, Path=Text}". Теперь все ок.

^ | ▾ • Ответить • Поделиться ›

**Катя** • месяц назад

Как динамически менять содержимое поля, в которое мы что-то вводим? по мере ввода добавлять символы и т д

^ | ▾ • Ответить • Поделиться ›

**Metanit** Модератор → Катя • месяц назад


делайте привязку к свойству Text и в конвертере меняйте значение

^ | ▾ • Ответить • Поделиться ›

ТАКЖЕ НА METANIT.COM

Vue.js | Props

2 комментария • 4 месяца назад

 **Metanit** — нет, и так должно работать, проверил в Google Chrome, MS Edge, Firefox


C# и .NET | Раннее и позднее связывание

3 комментария • 2 месяца назад

 **dev loop** — спасибо, ответили и на мой вопрос тоже)


Руководство по языку Go

2 комментария • месяц назад

 **Alexey** — О, спасибо большое, что уделили внимание этому языку

C++ | Динамические массивы

4 комментария • 2 месяца назад

 **Metanit** — в данном случае вы делаете ошибку, что переносите опыт работы с одним языком на другой. В C++ разных языках может быть разная терминология. В C++ Подписаться  Добавить Disqus на свой сайт [Добавить Disqus](#)  Конфиденциальность



[Вконтакте](#) | [Twitter](#) | [Google+](#) | [Канал сайта на youtube](#) | [Помощь сайту](#)

Copyright © metanit.com, 2012-2017. Все права защищены.