Tratamento de eventos no XAML

5 minutos

Depois de criar uma interface do usuário XAML, você pode adicionar código para responder às interações que ocorrem quando o usuário visita a página. O .NET MAUI notifica o aplicativo das entradas e interações do usuário por meio de eventos .NET padrão.

Nesta unidade, você aprenderá a manipular esses eventos e a executar as operações esperadas pelo usuário.

Nomenclatura dos elementos em uma página XAML

O código de manipulação de eventos muitas vezes precisa se referir a controles específicos e suas propriedades em uma página. Você pode atribuir um nome exclusivo a cada controle. Para fazer isso, use o atributo XAML `x:Name`. O atributo `x:Name` executa duas ações:

- Adiciona um campo privado ao arquivo code-behind gerado que é mapeado para esse elemento. Use esse campo no seu código para interagir com o elemento visual, definir as propriedades de runtime e manipular eventos.
- O XAML toma conhecimento do elemento por meio desse nome. Você pode consultar esses elementos a partir de outros elementos definidos no mesmo arquivo XAML.

Não é possível usar nenhuma cadeia de caracteres arbitrária ao nomear o elemento. O valor atribuído ao atributo `x:Name` é usado para criar um campo no código. Em vez disso, ele deve estar em conformidade com as convenções de nomenclatura para uma variável. O nome também deve ser exclusivo, pois é compilado para a definição de code-behind.

Depois de fornecer um nome para um elemento, você pode interagir com esse elemento no arquivo code-behind. O fragmento XAML a seguir define um controle `Label`. É chamado **CounterLabel** (esse exemplo foi tirado do aplicativo padrão gerado pelo modelo do .NET MAUI):

XML

```
<Label Text="Current count: 0"
    ...
```

```
        x:Name="CounterLabel"
        ... />
```

No code-behind dessa página, você pode mencionar esse controle por meio do campo `CounterLabel` e modificar suas propriedades:

C#

```
count++;
CounterLabel.Text = $"Current count: {count}";
```

> **Importante**
>
> O campo não será inicializado até que o método `InitializeComponent` da página seja executado. Esse método faz parte do processo de análise de XAML e instanciação de objetos. Coloque qualquer código que interaja com um elemento definido no XAML após esta chamada. A exceção a essa regra é a própria classe `ContentPage`. Você pode acessar todas as propriedades na classe antes de executar o método `InitializeComponent`. No entanto, se você definir as propriedades nessa classe no XAML, esses valores de propriedade substituirão os valores que você tiver definido antes de executar `InitializeComponent`.

Usar um atributo para conectar eventos

Muitos controles expõem propriedades que correspondem aos eventos aos quais esses controles podem responder, como o evento `Clicked` de um botão. Controles diferentes dão suporte a conjuntos de eventos variados. Por exemplo, um controle `Button` pode responder a eventos `Clicked`, `Pressed` e `Released`, enquanto um controle `Entry` tem eventos como `TextChanged`. Você pode inicializar uma propriedade de evento na marcação XAML de uma página e especificar o nome do método a ser executado quando o evento for disparado. O método de evento precisa atender aos seguintes requisitos de assinatura:

- Ele não pode retornar um valor; o método deve ser `void`.
- Ele deve ter dois parâmetros; uma referência `object` que indica o objeto que disparou o evento (conhecido como *remetente*) e um parâmetro `EventArgs` que contém todos os argumentos passados para o manipulador de eventos pelo remetente.
- O manipulador de eventos deve ser `private`. Isso não é implementado, mas se você tornar um manipulador de eventos público, ele ficará acessível para o mundo exterior e uma ação que não seja o evento esperado sendo disparado poderá invocá-lo.

- O manipulador de eventos poderá ser `async` se precisar executar operações assíncronas.

O exemplo a seguir mostra a definição do manipulador de eventos `Clicked` para o botão na amostra de aplicativo do modelo do .NET MAUI. O nome do método segue uma convenção padrão: **On**, seguido do nome do controle (o botão é chamado de **Counter**) e do nome do evento (**Clicked**). Essa convenção não é imposta, mas é uma boa prática:

C#

```
private void OnCounterClicked(object sender, EventArgs e)
{
    ...
}
```

Separação de interesses

Conectar eventos no XAML é conveniente, mas mistura o comportamento do controle com a definição da interface do usuário. Muitos desenvolvedores preferem manter esses elementos distintos e fazer todas as assinaturas do manipulador de eventos em code-behind para elementos nomeados. Esse método torna mais fácil ver o que está conectado e para onde o comportamento está sendo mapeado. Também torna mais difícil quebrar o código acidentalmente removendo um manipulador no XAML sem perceber. O compilador não irá captar um manipulador removido e só irá aparecer como um problema quando o código não desempenhar esse comportamento corretamente.

A opção por conectar manipuladores de eventos usando o XAML ou usando código é uma questão de preferência pessoal.

Para conectar um manipulador de eventos no código, use o operador `+=` para assinar o evento. Normalmente, você executa essa operação no construtor da página, após a chamada para `InitializeComponent`:

C#

```
public partial class MainPage : ContentPage, IPage
{
    public MainPage()
    {
        InitializeComponent();
        Counter.Clicked += OnCounterClicked;
    }

    ...

    private void OnCounterClicked(object sender, EventArgs e)
```

```
    {
        ...
    }
}
```

> **Observação**
>
> Você pode usar essa abordagem para assinar vários métodos de manipulação de eventos para o mesmo evento. Cada método de manipulação de eventos será executado quando o evento ocorrer, embora você não deva supor que será executado em uma ordem específica, então certifique-se de não introduzir nenhuma dependência entre eles.

Da mesma forma, você pode remover um manipulador de eventos cancelando a respectiva assinatura no evento com o operador `-=` mais à frente no seu aplicativo:

C#

```
Counter.Clicked -= OnCounterClicked;
```

Verificação de conhecimentos

1. O que são os parâmetros passados para um método de manipulação de eventos? *

- ○ O nome do evento e o nome do controle que gerou o evento.
- ○ Uma referência à página XAML e ao controle na página que gerou o evento
- ○ Uma referência ao controle que gerou o evento e a um objeto `EventArgs` que contém informações adicionais necessárias para processar o evento.

✔ **Esta é a resposta correta.**

2. Qual operador você pode usar para conectar um manipulador de eventos a um evento em C#? *

- ○ Usar o operador `+=`.

✔ **Esta é a resposta correta.**

- ○ Use o operador `subscribe`

Não é possível conectar a manipulação de eventos usando C#. Você precisa usar XAML.

Unidade seguinte: Exercício: criar sua primeira página em XAML

Continuar >