عنوان: **فراخوانی یک متداز یک کنترل WPF از XAML**
نویسنده: محبوبه محمدی
تاریخ: ۱۳۹۲/۰۳/۱۲ ۲۳:۰
آدرس: www.dotnettips.info
برچسب‌ها: WPF

در بعضی مواقع نیاز است که یک متد از یک کنترل درون XAML فراخوانی شود. برای مثال لازم است یکی از متدهای یک کنترل در یک استایل فراخوانی شود. یکی از روش‌های انجام این کار استفاده از خصوصیت‌های پیوست شده(AttachedPropery) است. شیوه‌ی کار به این صورت است که یک خصوصیت از نوع Bool ایجاد می‌کنیم. هنگامیکه مقدار این خصوصیت تغییر کند یک رویه فراخوانی می‌شود که کار فراخوانی متد مورد نظر را انجام میدهد:

```
public class SelectAllBehavior
    {
        public static bool GetSelectAll(TextBoxBase target)
        {
            return (bool)target.GetValue(SelectAllProperty);
        }

        public static void SetSelectAll(TextBoxBase target, bool value)
        {
            target.SetValue(SelectAllProperty, value);
        }

        public static readonly DependencyProperty SelectAllProperty =
DependencyProperty.RegisterAttached("SelectAll", typeof(bool), typeof(SelectAllBehavior), new
UIPropertyMetadata(false, OnSelectAllPropertyChanged));

        static void OnSelectAllPropertyChanged(DependencyObject o, DependencyPropertyChangedEventArgs
e)
        {
            ((TextBoxBase)o).SelectAll();
        }
    }
```

برای استفاده از کلاس فوق درون یک استایل باید به شکل زیر عمل کرد:

```
<Style TargetType="{x:Type TextBoxBase}" >
    <Style.Triggers>
        <Trigger Property="IsFocused" Value="True">
            <Setter Property="x: SelectAllBehavior.SelectAll" Value="True"/>
        </Trigger>
    </Style.Triggers>
</Style>
```

در تکه کد بالا ،هنگامی که خصوصیت IsFocused مربوط به کنترل TextBox برابر True می‌شود، یعنی Focus روی TextBox قرار می‌گیرد،مقدار خصوصیت پیوست شده نیز برابر True می‌شود که همانطور که گفته شد باعث فراخوانی OnSelectAllPropertyChanged می‌شود.

تا اینجا فراخوانی یک متد از کنترل از طریق استایل توضیح داده شد، همانطور که در عنوان مطلب آورده شده است. اما اگر بخواهید مثال فوق را به درستی اجرا کنید یعنی هنگام کلیک روی TextBox متن درون آن انتخاب شود، نیاز به اضافه کردن کدهای دیگری وجود دارد. چرا که به صورت پیش فرض زمانی که MouseLeftButtonUp اجرا می‌شود در صورتی که حالت متن به صورت انتخاب شده باشد، از حالت انتخاب خارج میشود. این بار برای اینکار از خصوصیت‌های پیوست شده(AttachedPropery) برای کنترل رویداد PreviewMouseLeftButtonDown استفاده میکنیم و هنگام فشرده شدن کلید سمت چپ موس رویدادهای Click و بقیه LeftMouseButtonDownها را غیرفعال می‌کنیم.

```
public class PreviewMouseLeftButtonDownBehavior
{
    public static readonly DependencyProperty PreviewMouseLeftButtonDownProperty =
DependencyProperty.RegisterAttached("PreviewMouseLeftButtonDown"
                  , typeof(bool?)
                  , typeof(PreviewMouseLeftButtonDownBehavior)
                  , new UIPropertyMetadata(null, OnPreviewMouseLeftButtonDown)
```

```
                );

    public static void SetPreviewMouseLeftButtonDown(DependencyObject target, bool? value)
    {
        target.SetValue(PreviewMouseLeftButtonDownProperty, value);
    }
    public static object GetPreviewMouseLeftButtonDown(DependencyObject target)
    {
        return target.GetValue(PreviewMouseLeftButtonDownProperty);
    }

    public static void OnPreviewMouseLeftButtonDown(DependencyObject obj,
DependencyPropertyChangedEventArgs e)
    {
        var control = obj as Control;
        if (control == null)
            return;
        if (e.NewValue != null && e.OldValue == null)
            control.PreviewMouseLeftButtonDown += control_PreviewMouseLeftButtonUp;
        else if ((e.NewValue == null) && (e.OldValue != null))
        {
            control.PreviewMouseLeftButtonDown -= control_PreviewMouseLeftButtonUp;
        }
    }
    static void control_PreviewMouseLeftButtonUp(object sender,
System.Windows.Input.MouseButtonEventArgs e)
    {
        var tb = (sender as TextBox);
        if (tb != null)
        {
            if (!tb.IsKeyboardFocusWithin)
            {
                e.Handled = true;
                tb.Focus();
            }
        }
    }
}
```

و استایل را به صورت زیر تغییر میدهیم:

```
<Style TargetType="{x:Type TextBoxBase}" >
    <Setter Property="x:PreviewMouseLeftButtonDownBehavior.IsPreviewMouseLeftButtonDown" Value="True"/>
    <Style.Triggers>
        <Trigger Property="IsKeyboardFocusWithin" Value="True">
            <Setter Property="InputLanguageManager.InputLanguage" Value="fa-ir" />
        </Trigger>
        <Trigger Property="IsFocused" Value="True">
            <Setter Property="xamlServices:SelectAllTextBoxBehavior.SelectAll" Value="True"/>
        </Trigger>
    </Style.Triggers>
</Style>
```

در این مثال با Focus رو هر TextBox که استایل فوق را به کار گرفته باشد، متن در حالت انتخاب شده قرار میگیرد. (البته این مشروط به این است که نیاز نباشد رویداد PreviewMouseLeftButtonDown کنترل مورد نظر درون برنامه مقیدسازی(Bind) شود.)