



```
private bool HandleBackspaceKey()
{
    ReplaceVirtualSpaces();

    var snapshot = TextView.TextBuffer.CurrentSnapshot;
    var caretPos = Caret.Position.BufferPosition.Position;

    // Determine the number of spaces until the previous tab stop.
    var spacesToRemove = ((Caret.Column - 1) % IndentSize) + 1;

    // Make sure we only delete spaces.
    for (var i = 0; i < spacesToRemove; i++)
    {
        var snapshotPos = caretPos - 1 - i;
        if (snapshotPos < 0 || snapshot[snapshotPos] != ' ')
        {
            spacesToRemove = i;
            break;
        }
    }

    if (spacesToRemove > 1)
    {
        TextView.TextBuffer.Delete(new Span(caretPos - spacesToRemove,
spacesToRemove));
        return true;
    }
    else
    {
        return false;
    }
}

private bool HandleDeleteKey()
{
    // If we are in virtual space, we should already be at the end of the line,
```

```

// so let Visual Studio handle the keypress.
if (Caret.InVirtualSpace)
{
    return false;
}

var snapshot = TextView.TextBuffer.CurrentSnapshot;
var caretPos = Caret.Position.BufferPosition.Position;

// Determine the number of spaces until the next tab stop.
var spacesToRemove = IndentSize - (CaretColumn % IndentSize);

// Make sure we only delete spaces.
for (var i = 0; i < spacesToRemove; i++)
{
    var snapshotPos = caretPos + i;
    if (snapshotPos >= snapshot.Length || snapshot[snapshotPos] != ' ')
    {
        spacesToRemove = i;
        break;
    }
}

if (spacesToRemove > 1)
{
    TextView.TextBuffer.Delete(new Span(caretPos, spacesToRemove));
    return true;
}
else
{
    return false;
}
}

private void ReplaceVirtualSpaces()
{
    if (Caret.InVirtualSpace)
    {
        TextView.TextBuffer.Insert(Caret.Position.BufferPosition, new string(' ',

```

```
Caret.Position.VirtualSpaces));  
    Caret.MoveTo(CaretLine.End);  
}  
}
```