Generando imágenes (códigos de Barras) en Reporting Services

Por **FREDDY LEANDRO ANGARITA C.  
SqlServer MVP**   
[Perfil MVP](https://mvp.support.microsoft.com/es-es/mvp/Freddy%20Leandro%20Angarita%20Castellanos-4028407)   
  
[freddy\_angarita@hotmail.com](mailto:freddy_angarita@hotmail.com)   
<http://geeks.ms/blogs/fangarita/default.aspx>

Una de las principales características de Reporting Services e Integration Services de  Microsoft es la habilidad de agregar código .Net  en diferentes puntos de los items de la solución

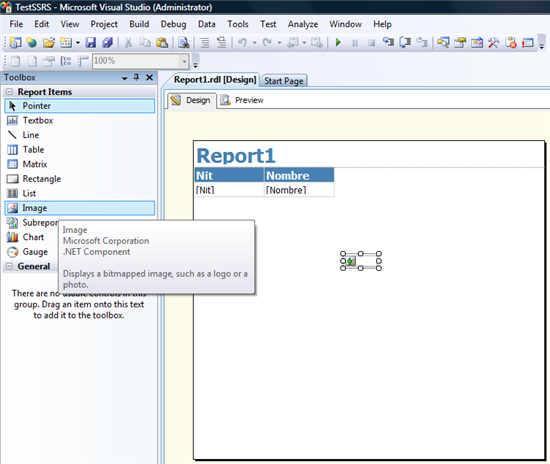
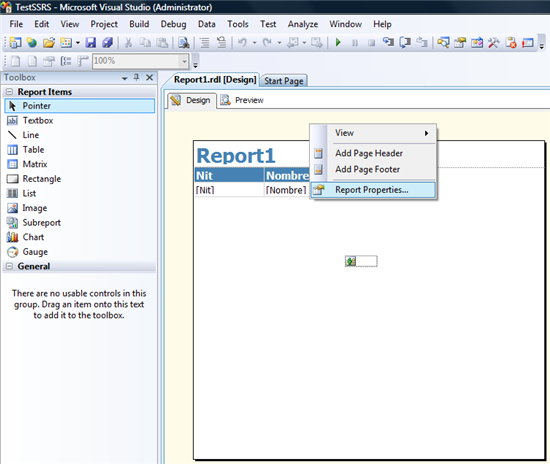
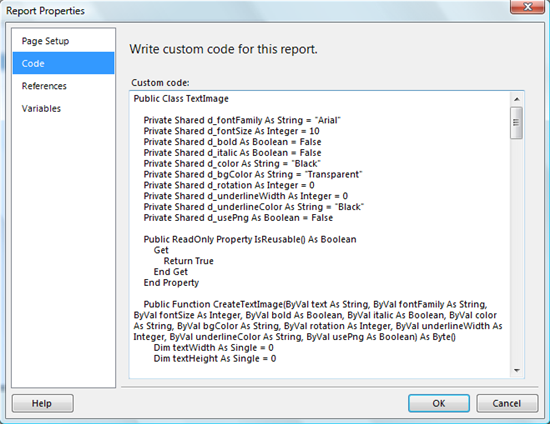
En éste artículo veremos cómo generar una imagen dinámicamente directamente dentro de un reporte, sin la necesidad de agregar ningún código de terceros, ni de agregar assemblies de otros proveedores, sólo código .Net directamente integrado al reporte

**El problema  
Dentro de los diferentes escenarios que se tienen para la generación de imágenes dentro de un reporte el que me llevó a desarrollar ésta solución fue la solicitud de un cliente de generar códigos de barras, y que los mismos fueran integrados al reporte, entonces el primer impulso que se tiene es generar una imagen con un manejador HTTP ver artículo de codeproject Text To Image With Rotation For The Web ésta es una buena solución, de hecho buena parte del código presentado en éste artículo fue extraído de éste artículo, traducido a Visual Basic y adaptado para funcionar en Reporting Services**

**La Solución (Primera versión)  
Crear una página en donde mediante parámetros querystring se generara una imagen mediante la respuesta HTTP creada por el ashx y puesta como una imagen normal dentro del reporte, esto presenta el problema que cuando el usuario pierde la conexión, no podrá ver la imagen de nuevo y la misma no haría parte de la respuesta generada por reporting services sino que sería una referencia a una imagen por demás debía ser una página abierta al público sin contexto de seguridad lo cual sería problemático y se prestaría para posibles inconvenientes**

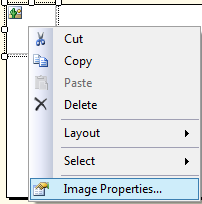
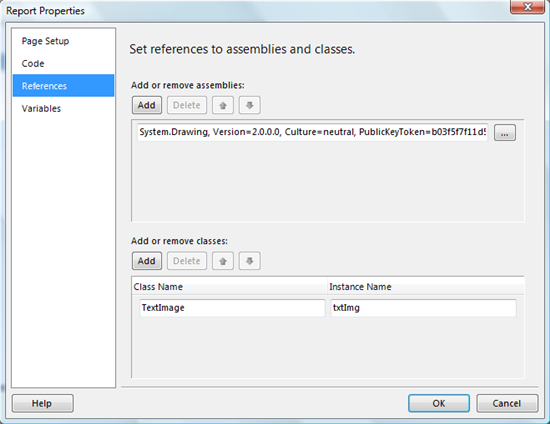
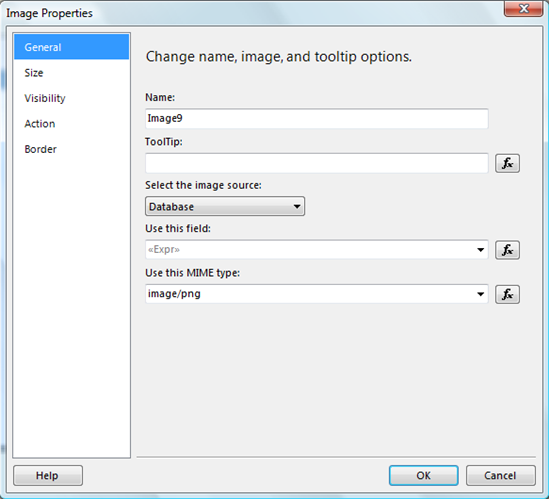
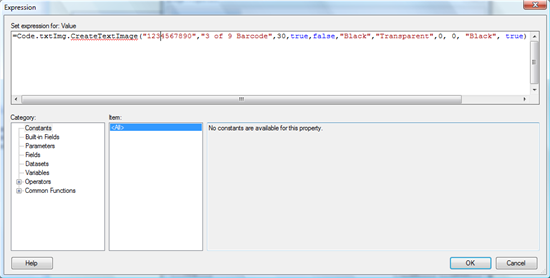
**La Solución Reporting Services  
Usando la característica que permite agregar código a un reporte, se agregó una clase, si una clase, al código personalizado del mismo y luego llamando éste código desde el mismo reporte se pudo generar la imagen para el código de barras  
Los pasos a seguir para elaborar ésta solución son:**

1. Agregar una imagen al reporte en cualquier parte del mismo, incluyendo un grid, o un el área de diseño del reporte

* 1.   
        
     2. Hacer Clic fuera del recuadro y seleccionar propiedades del reporte
  2.   
        
     3. Cuando aparezca el diálogo se hace clic sobre el tab de código y se procede a copiar o a digitar la clase que necesitamos para la generación de la imagen a partir de un texto
  3.   
        
     El código debe ser algo como esto
  4. Public Class TextImage
  5. Private Shared d\_fontFamily As String = "Arial"
  6. Private Shared d\_fontSize As Integer = 10
  7. Private Shared d\_bold As Boolean = False
  8. Private Shared d\_italic As Boolean = False
  9. Private Shared d\_color As String = "Black"
  10. Private Shared d\_bgColor As String = "Transparent"
  11. Private Shared d\_rotation As Integer = 0
  12. Private Shared d\_underlineWidth As Integer = 0
  13. Private Shared d\_underlineColor As String = "Black"
  14. Private Shared d\_usePng As Boolean = False
  15. Public ReadOnly Property IsReusable() As Boolean
  16. Get
  17. Return True
  18. End Get
  19. End Property
  20. Public Function CreateTextImage(ByVal text As String, ByVal fontFamily As String, ByVal fontSize As Integer, ByVal bold As Boolean, ByVal italic As Boolean, ByVal color As String, ByVal bgColor As String, ByVal rotation As Integer, ByVal underlineWidth As Integer, ByVal underlineColor As String, ByVal usePng As Boolean) As Byte()
  21. Dim textWidth As Single = 0
  22. Dim textHeight As Single = 0
  23. Dim fontStyle As System.Drawing.FontStyle = System.Drawing.FontStyle.Regular
  24. If bold Then fontStyle = fontStyle Or Drawing.FontStyle.Bold
  25. If italic Then fontStyle = fontStyle Or Drawing.FontStyle.Italic
  26. Using font As New System.Drawing.Font(fontFamily, fontSize, fontStyle)
  27. ' Get the text measurements.
  28. ' I need to create a dummy bitmap so that I can get a Graphics object to get the text measurements.
  29. Using bitmap As New System.Drawing.Bitmap(1, 1)
  30. Using graphics As System.Drawing.Graphics = System.Drawing.Graphics.FromImage(bitmap)
  31. textWidth = graphics.MeasureString(text, font).Width
  32. textHeight = graphics.MeasureString(text, font).Height
  33. End Using
  34. End Using
  35. -
  36. ' Calculate the needed bitmap measurements based on the text measurements.
  37. Dim bitmapWidth As Integer = GetRotatedRectangleWidth(textWidth, textHeight, rotation)
  38. Dim bitmapHeight As Integer = GetRotatedRectangleHeight(textWidth, textHeight, rotation)
  39. ' Now I create the real bitmap of the necessary size to fit the text.
  40. Using bitmap As New System.Drawing.Bitmap(bitmapWidth, bitmapHeight)
  41. Using graphics As System.Drawing.Graphics = System.Drawing.Graphics.FromImage(bitmap)
  42. If (Not usePng) Then graphics.Clear(System.Drawing.Color.FromArgb(255, 255, 255, 204)) 'Color.FromArgb(&HFF, &HFF, &HFF, &HCC)
  43. ' Since I will be rotating text, I need to move the rotation point using the TranslateTransform.
  44. Dim x As Integer, y As Integer
  45. ' But first I need to know the location of the rotation point.
  46. GetXY(rotation, textWidth, textHeight, x, y)
  47. graphics.TranslateTransform(x, y)
  48. ' Now rotate and draw the text.
  49. graphics.RotateTransform(rotation)
  50. ' Fill in the background color
  51. Using brush As New System.Drawing.SolidBrush(System.Drawing.ColorTranslator.FromHtml(bgColor))
  52. graphics.SmoothingMode = System.Drawing.Drawing2D.SmoothingMode.HighQuality
  53. graphics.FillRectangle(brush, 0, 0, textWidth, textHeight)
  54. End Using
  55. graphics.TextRenderingHint = Drawing.Text.TextRenderingHint.AntiAlias
  56. Using brush As New System.Drawing.SolidBrush(System.Drawing.ColorTranslator.FromHtml(color))
  57. graphics.DrawString(text, font, brush, 0, 0)
  58. End Using
  59. If (underlineWidth > 0) Then
  60. graphics.SmoothingMode = System.Drawing.Drawing2D.SmoothingMode.HighQuality
  61. Using pen As New System.Drawing.Pen(System.Drawing.ColorTranslator.FromHtml(underlineColor), underlineWidth)
  62. graphics.DrawLine(pen, 0, textHeight, textWidth, textHeight)
  63. End Using
  64. End If
  65. graphics.Flush()
  66. Dim m As New System.IO.MemoryStream()
  67. If usePng Then
  68. bitmap.Save(m, System.Drawing.Imaging.ImageFormat.Png)
  69. Else
  70. bitmap.Save(m, System.Drawing.Imaging.ImageFormat.Gif)
  71. End If
  72. If (usePng) Then
  73. Return m.ToArray()
  74. Else
  75. ' transparency hack.
  76. Dim n As Byte() = {}
  77. n = m.ToArray()
  78. n(787) = 254
  79. Return n
  80. End If
  81. End Using ' using graphics
  82. End Using ' using bitmap
  83. End Using ' using font
  84. End Function
  85. Private Sub GetXY(ByVal rotation As Integer, ByVal tw As Single, ByVal th As Single, ByRef xT As Integer, ByRef yT As Integer)
  86. xT = 0
  87. yT = 0
  88. Dim radians As Double = GetRadians(GetReferenceAngleForPositioning(rotation))
  89. If (rotation >= 0 AndAlso rotation <= 90) Then
  90. xT = Convert.ToInt32(Math.Sin(radians) \* th)
  91. ElseIf (rotation > 90 AndAlso rotation <= 180) Then
  92. xT = Convert.ToInt32((Math.Sin(radians) \* tw) + (Math.Cos(radians) \* th))
  93. yT = Convert.ToInt32(Math.Sin(radians) \* th)
  94. ElseIf (rotation > 180 AndAlso rotation <= 270) Then
  95. xT = Convert.ToInt32(Math.Cos(radians) \* tw)
  96. yT = Convert.ToInt32((Math.Cos(radians) \* th) + (Math.Sin(radians) \* tw))
  97. Else
  98. yT = Convert.ToInt32(Math.Sin(radians) \* tw)
  99. End If
  100. End Sub
  101. Private Function GetRotatedRectangleWidth(ByVal width As Single, ByVal height As Single, ByVal rotation As Integer) As Integer
  102. Return GetRotatedRectangleWidth(width, height, GetRadians(GetReferenceAngleForSizing(rotation)))
  103. End Function
  104. Private Function GetRotatedRectangleWidth(ByVal width As Single, ByVal height As Single, ByVal rotationInRadians As Double) As Integer
  105. Dim w1 As Double = width \* Math.Cos(rotationInRadians)
  106. Dim w2 As Double = height \* Math.Sin(rotationInRadians)
  107. Return Convert.ToInt32(Math.Ceiling(w1 + w2))
  108. End Function
  109. Private Function GetRotatedRectangleHeight(ByVal width As Single, ByVal height As Single, ByVal rotation As Integer) As Integer
  110. Return GetRotatedRectangleHeight(width, height, GetRadians(GetReferenceAngleForSizing(rotation)))
  111. End Function
  112. Private Function GetRotatedRectangleHeight(ByVal width As Single, ByVal height As Single, ByVal rotationInRadians As Double)
  113. Dim h1 As Double = width \* Math.Sin(rotationInRadians)
  114. Dim h2 As Double = height \* Math.Cos(rotationInRadians)
  115. Return Convert.ToInt32(Math.Ceiling(h1 + h2))
  116. End Function
  117. Private Function GetRadians(ByVal referenceAngle As Double) As Double
  118. Return Math.PI \* referenceAngle / 180.0
  119. End Function
  120. ' The only difference between the reference angles for sizing and positioning is in the 3rd quadrant (> 90 <= 180).
  121. Private Function GetReferenceAngleForSizing(ByVal rotationInDegrees As Integer) As Double
  122. If (rotationInDegrees >= 0 And rotationInDegrees <= 90) Then
  123. Return rotationInDegrees
  124. ElseIf (rotationInDegrees > 90 And rotationInDegrees <= 180) Then
  125. Return 180 - rotationInDegrees
  126. ElseIf (rotationInDegrees > 180 And rotationInDegrees <= 270) Then
  127. Return rotationInDegrees - 180
  128. ElseIf (rotationInDegrees > 270 And rotationInDegrees <= 360) Then
  129. Return 360 - rotationInDegrees
  130. Else
  131. Return rotationInDegrees
  132. End If
  133. End Function
  134. Private Function GetReferenceAngleForPositioning(ByVal rotationInDegrees As Integer) As Double
  135. If (rotationInDegrees >= 0 And rotationInDegrees <= 90) Then
  136. Return rotationInDegrees
  137. ElseIf (rotationInDegrees > 90 And rotationInDegrees <= 180) Then
  138. Return rotationInDegrees - 90
  139. ElseIf (rotationInDegrees > 180 And rotationInDegrees <= 270) Then
  140. Return rotationInDegrees - 180
  141. ElseIf (rotationInDegrees > 270 And rotationInDegrees <= 360) Then
  142. Return 360 - rotationInDegrees
  143. Else
  144. Return rotationInDegrees
  145. End If
  146. End Function
  147. End Class

Claro, se ve un poco largo, pero tiene varias funcionalidades, entre ellas ver una imagen rotada también

4. Luego, en el mismo dialogo se agregan las referencias .Net que el código agregado va a utilizar, en éste caso System.Drawing. Adicionalmente, se agrega la identificación de la Clase y su nombre de instancia (SSRS instancia automáticamente la clase ya que los métodos utilizados son de instancia) para usarla como referencia en el código

* 1. 
  2.   
         
     Una vez finalizado se hace clic en ok  
     5. En el diseñador del reporte se oprime botón derecho sobre la imagen que se agregó, se hace clic sobre propiedades y aparece el diálogo, y debemos configurarla como aparece en la imagen
  3.   
        
     6. En el diálogo en donde indica "Usar éste campo" o "Use this field" hace clic sobre el botón fx, el cual presenta el siguiente diálogo, en donde digitaremos la expresión que aparece en la imagen la cual realizará el llamado al código enviado como parámetros: el texto a convertir, el tipo de letra a usar, el tamaño de la misma, si es en negrilla, si es itálica, color de la letra, color de fondo de la letra,  ancho del subrayado, color del subrayado y por último si utiliza el formato png como formato de salida
  4. 

Hacemos clic en Ok y ya tenemos configurada la imagen  
7. Una vez finalizado el proceso podremos generar imágenes directamente en el reporte basado en texto como se aprecia en la imagen

## Observaciones

Para generar códigos de barra es requerido tener un True Type Font en el mismo computador en donde está instalado Reporting Services, algunos de los tipos de letra que generan códigos de barra son licenciados buscando parece que hay algunos que son gratis pero recomiendo leer detenidamente la licencia y ver sus condiciones de uso antes de ponerlo en algún ambiente de producción

El reporte adjunto probablemente no funcione al primer intento ya que intenta leer de una base de datos llamada prueba con datos de ejemplo, es posible eliminar la tabla y ver solamente el código de barras

Otra situación que se presentará es que el código de barras posiblemente no sea muy claro y el mismo no sea legible por algunos dispositivos, razón por la cual publicaré la [parte 2](http://geeks.ms/blogs/fangarita/archive/2010/10/12/webcast.aspx) de éste artículo  
   
   
**FREDDY LEANDRO ANGARITA CASTELLANOS**SQLSERVER MVP