عنوان: **معرفی یک ابزار گزارشگیری رایگان مخصوص WPF**
نویسنده: وحید نصیری
تاریخ: ۱۳۸۹/۰۸/۰۱ ۱۴:۵۷:۰۰
آدرس: www.dotnettips.info
برچسب‌ها: MVVM

تا صحبت از گزارشگیری به میان بیاید احتمالا معرفی ابزارهای تجاری مانند Reporting services ، کریستال ریپورت، fast-report.com ، stimulsoft.com و امثال آن درصدر لیست توصیه کنندگان و مشاوران قرار خواهند داشت. اما خوب برای ایجاد یک گزارشگیری ساده حتما نیازی نیست تا به این نوع ابزارهای تجاری مراجعه کرد. ابزار رایگان و سورس باز جالبی هم در این باره جهت پروژه‌های WPF در دسترس است:

Open-Source .NET WPF Reporting Engine

در ادامه در طی یک مثال قصد داریم از این کتابخانه استفاده کنیم:

1) تنظیم وابستگی‌ها
پس از دریافت کتابخانه فوق، ارجاعات زیر باید به پروژه شما اضافه شوند:
CodeReason.Reports.dll (از پروژه فوق) و ReachFramework.dll (جزو اسمبلی‌های استاندارد دات نت است)

2) تهیه منبع داده گزارش
کتابخانه‌ی فوق به صورت پیش فرض با DataTable کار می‌کند. بنابراین کوئری‌های شما یا باید خروجی DataTable داشته باشد یا باید از یک سری extension methods برای تبدیل IEnumerable به DataTable استفاده کرد (در پروژه پیوست شده در پایان مطلب، این موارد موجود است).
برای مثال فرض کنید می‌خواهیم رکوردهایی را از نوع کلاس Product زیر در گزارش نمایش دهیم:

```
namespace WpfRptTests.Model
{
  public class Product
  {
      public string Name { set; get; }
      public int Price { set; get; }
  }
}
```

3) تعریف گزارش
الف) اضافه کردن فایل تشکیل دهنده ساختار و ظاهر گزارش
گزارش‌های این کتابخانه مبتنی است بر اشیاء FlowDocument استاندارد WPF . بنابراین از منوی پروژه گزینه‌ی Add new item در قسمت WPF آن یک FlowDocument جدید را به پروژه اضافه کنید (باید دقت داشت که Build action این فایل باید به Content تنظیم گردد). ساختار ابتدایی این FlowDocument به صورت زیر خواهد بود که به آن FlowDirection و FontFamily مناسب جهت گزارشات فارسی اضافه شده است. همچنین فضای نام مربوط به کتابخانه‌ی گزارشگیری CodeReason.Reports نیز باید اضافه گردد.

```
<FlowDocument xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
            xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
            FlowDirection="RightToLeft" FontFamily="Tahoma"
            xmlns:xrd="clr-namespace:CodeReason.Reports.Document;assembly=CodeReason.Reports"
            PageHeight="29.7cm" PageWidth="21cm" ColumnWidth="21cm">

</FlowDocument>
```

مواردی که در ادامه ذکر خواهند شد محتوای این گزارش را تشکیل می‌دهند:

ب) مشخص سازی خواص گزارش

```
<xrd:ReportProperties>
      <xrd:ReportProperties.ReportName>SimpleReport</xrd:ReportProperties.ReportName>
      <xrd:ReportProperties.ReportTitle>گزارش از محصولات</xrd:ReportProperties.ReportTitle>
  </xrd:ReportProperties>
```

در اینجا ReportName و ReportTitle باید مقدار دهی شوند (دو dependency property که در کتابخانه‌ی CodeReason.Reports تعریف شده‌اند)

ج) مشخص سازی Page Header و Page Footer

اگر می‌خواهید عباراتی در بالا و پایین تمام صفحات گزارش تکرار شوند می‌توان از SectionReportHeader و SectionReportFooter این کتابخانه به صورت زیر استفاده کرد:

```
<xrd:SectionReportHeader PageHeaderHeight="2" Padding="10,10,10,0" FontSize="12">
      <Table CellSpacing="0">
          <Table.Columns>
              <TableColumn Width="*" />
              <TableColumn Width="*" />
          </Table.Columns>
          <TableRowGroup>
              <TableRow>
                  <TableCell>
                      <Paragraph>
                          <xrd:InlineContextValue PropertyName="ReportTitle" />
                      </Paragraph>
                  </TableCell>
                  <TableCell>
                      <Paragraph TextAlignment="Right">
                          <xrd:InlineDocumentValue PropertyName="PrintDate" Format="dd.MM.yyyy
HH:mm:ss" />
                      </Paragraph>
                  </TableCell>
              </TableRow>
          </TableRowGroup>
      </Table>
  </xrd:SectionReportHeader>

  <xrd:SectionReportFooter PageFooterHeight="2" Padding="10,0,10,10" FontSize="12">
      <Table CellSpacing="0">
          <Table.Columns>
              <TableColumn Width="*" />
              <TableColumn Width="*" />
          </Table.Columns>
          <TableRowGroup>
              <TableRow>
                  <TableCell>
                      <Paragraph>
                          نام کاربر:
                          <xrd:InlineDocumentValue PropertyName="RptBy" Format="dd.MM.yyyy HH:mm:ss" />
                      </Paragraph>
                  </TableCell>
                  <TableCell>
                      <Paragraph TextAlignment="Right">
                          صفحه
                          <xrd:InlineContextValue PropertyName="PageNumber" FontWeight="Bold" /> از
                          <xrd:InlineContextValue PropertyName="PageCount" FontWeight="Bold" />
                      </Paragraph>
                  </TableCell>
              </TableRow>
          </TableRowGroup>
      </Table>
  </xrd:SectionReportFooter>
```

دو نکته در اینجا حائز اهمیت هستند: xrd:InlineDocumentValue و xrd:InlineContextValue

InlineDocumentValue را می‌توان در کدهای برنامه به صورت سفارشی اضافه کرد. بنابراین هر جایی که نیاز بود مقدار ثابتی از

طریق کد نویسی به گزارش تزریق و اضافه شود می‌توان از InlineDocumentValue استفاده کرد. برای مثال در کدهای ViewModel برنامه که در ادامه ذکر خواهد شد دو مقدار PrintDate و RptBy به صورت زیر تعریف و مقدار دهی شده‌اند:

```
data.ReportDocumentValues.Add("PrintDate", DateTime.Now);
data.ReportDocumentValues.Add("RptBy", "وحید");
```

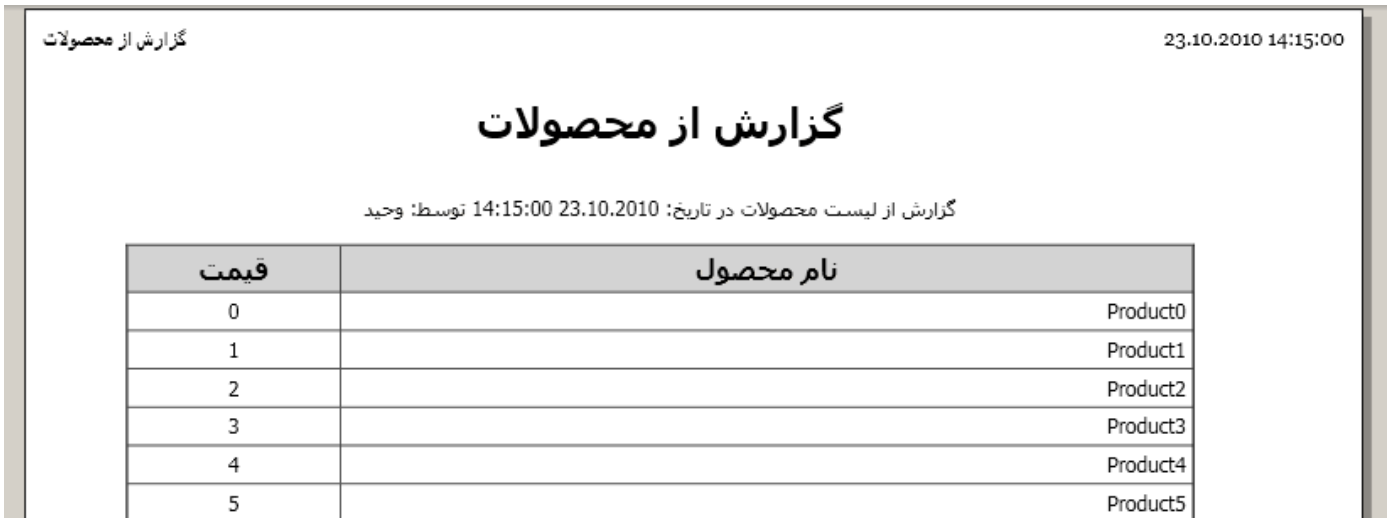
برای مشاهده مقادیر مجاز مربوط به InlineContextValue به فایل ReportContextValueType.cs سورس کتابخانه مراجعه کنید که شامل PageNumber, PageCount, ReportName, ReportTitle است و توسط CodeReason.Reports به صورت پویا تنظیم خواهد شد.

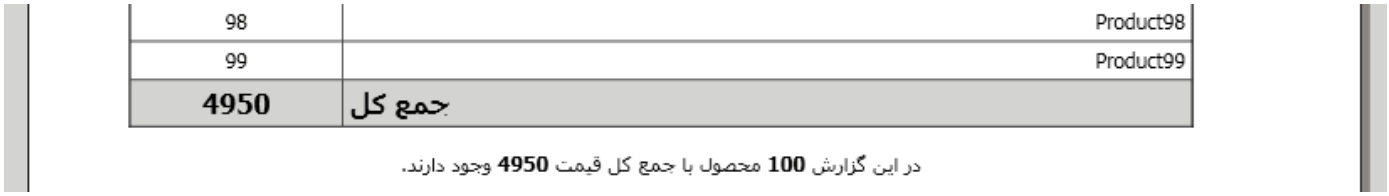
د) مشخص سازی ساختار تولیدی گزارش

```
<Section Padding="80,10,40,10" FontSize="12">
      <Paragraph FontSize="24" TextAlignment="Center" FontWeight="Bold">
          <xrd:InlineContextValue PropertyName="ReportTitle" />
      </Paragraph>
      <Paragraph  TextAlignment="Center">
          گزارش از لیست محصولات در تاریخ:
          <xrd:InlineDocumentValue PropertyName="PrintDate" Format="dd.MM.yyyy HH:mm:ss" />
          توسط:
          <xrd:InlineDocumentValue PropertyName="RptBy" Format="dd.MM.yyyy HH:mm:ss" />
      </Paragraph>
      <xrd:SectionDataGroup DataGroupName="ItemList">
          <Table CellSpacing="0" BorderBrush="Black" BorderThickness="0.02cm">
              <Table.Resources>
                  <!-- Style for header/footer rows. -->
                  <Style x:Key="headerFooterRowStyle" TargetType="{x:Type TableRowGroup}">
                      <Setter Property="FontWeight" Value="DemiBold"/>
                      <Setter Property="FontSize" Value="16"/>
                      <Setter Property="Background" Value="LightGray"/>
                  </Style>

                  <!-- Style for data rows. -->
                  <Style x:Key="dataRowStyle" TargetType="{x:Type TableRowGroup}">
                      <Setter Property="FontSize" Value="12"/>
                  </Style>

                  <!-- Style for data cells. -->
                  <Style TargetType="{x:Type TableCell}">
                      <Setter Property="Padding" Value="0.1cm"/>
                      <Setter Property="BorderBrush" Value="Black"/>
                      <Setter Property="BorderThickness" Value="0.01cm"/>
                  </Style>
              </Table.Resources>

              <Table.Columns>
                  <TableColumn Width="0.8*" />
                  <TableColumn Width="0.2*" />
              </Table.Columns>
              <TableRowGroup Style="{StaticResource headerFooterRowStyle}">
                  <TableRow>
                      <TableCell>
                          <Paragraph TextAlignment="Center">
                              <Bold>نام محصول</Bold>
                          </Paragraph>
                      </TableCell>
                      <TableCell>
                          <Paragraph TextAlignment="Center">
                              <Bold>قیمت</Bold>
                          </Paragraph>
                      </TableCell>
                  </TableRow>
              </TableRowGroup>

              <TableRowGroup Style="{StaticResource dataRowStyle}">
                  <xrd:TableRowForDataTable TableName="Product">
                      <TableCell>
                          <Paragraph>
                              <xrd:InlineTableCellValue PropertyName="Name" />
                          </Paragraph>
                      </TableCell>
                      <TableCell>
                          <Paragraph TextAlignment="Center">
                              <xrd:InlineTableCellValue PropertyName="Price" AggregateGroup="Group1" />
```

```
                        </Paragraph>
                    </TableCell>
                </xrd:TableRowForDataTable>
            </TableRowGroup>

            <TableRowGroup Style="{StaticResource headerFooterRowStyle}">
                <TableRow>
                    <TableCell>
                        <Paragraph TextAlignment="Right">
                            <Bold>جمع کل</Bold>
                        </Paragraph>
                    </TableCell>
                    <TableCell>
                        <Paragraph TextAlignment="Center">
                            <Bold>
                                <xrd:InlineAggregateValue AggregateGroup="Group1"
                                                          AggregateValueType="Sum"
                                                          EmptyValue="0"
                                                          FontWeight="Bold" />
                            </Bold>
                        </Paragraph>
                    </TableCell>
                </TableRow>
            </TableRowGroup>

        </Table>

        <Paragraph TextAlignment="Center" Margin="5">
            در این گزارش
            <xrd:InlineAggregateValue AggregateGroup="Group1"
                                      AggregateValueType="Count"
                                      EmptyValue="هیچ"
                                      FontWeight="Bold" /> محصول با جمع کل قیمت
            <xrd:InlineAggregateValue AggregateGroup="Group1"
                                      AggregateValueType="Sum"
                                      EmptyValue="0"
                                      FontWeight="Bold" /> وجود دارند.
        </Paragraph>
    </xrd:SectionDataGroup>
</Section>
```

برای اینکه بتوان این قسمت‌ها را بهتر توضیح داد، نیاز است تا تصاویر مربوط به خروجی این گزارش نیز ارائه شوند:





Product98	98
Product99	99
جمع کل	4950

در این گزارش **100** محصول با جمع کل قیمت **4950** وجود دارند.

در ابتدا توسط دو پاراگراف، عنوان گزارش و یک سطر زیر آن نمایش داده شده‌اند. بدیهی است هر نوع شیء و فرمت مجاز در FlowDocument را می‌توان در این قسمت نیز قرار داد.
سپس یک SectionDataGroup جهت نمایش لیست آیتم‌ها اضافه شده و داخل آن یک جدول که بیانگر ساختار جدول نمایش رکوردهای گزارش می‌باشد، ایجاد گردیده است.
سه TableRowGroup در این جدول تعریف شده‌اند.
TableRowGroup های اولی و آخری دو سطر اول و آخر جدول گزارش را مشخص می‌کنند (سطر عناوین ستون‌ها در ابتدا و سطر جمع کل در پایان گزارش)
از TableRowGroup میانی برای نمایش رکوردهای مرتبط با نام جدول مورد گزارشگیری استفاده شده است. توسط TableRowForDataTable آن نام این جدول باید مشخص شود که در اینجا همان نام کلاس مدل برنامه است. به کمک InlineTableCellValue، خاصیت‌هایی از این کلاس را که نیاز است در گزارش حضور داشته باشند، ذکر خواهیم کرد. نکته‌ی مهم آن AggregateGroup ذکر شده است. توسط آن می‌توان اعمال جمع، محاسبه تعداد، حداقل و حداکثر و امثال آن‌را که در فایل InlineAggregateValue.cs سورس کتابخانه ذکر شده‌اند، به فیلدهای مورد نظر اعمال کرد. برای مثال می‌خواهیم جمع کل قیمت را در پایان گزارش نمایش دهیم به همین جهت نیاز بود تا یک AggregateGroup را برای این منظور تعریف کنیم.
از این AggregateGroup در سومین TableRowGroup تعریف شده به کمک xrd:InlineAggregateValue جهت نمایش جمع نهایی استفاده شده است.
همچنین اگر نیاز بود در پایان گزارش اطلاعات بیشتری نیز نمایش داده شود به سادگی می‌توان با تعریف یک پاراگراف جدید، اطلاعات مورد نظر را نمایش داد.

4) نمایش گزارش تهیه شده
نمایش این گزارش بسیار ساده است. View برنامه به صورت زیر خواهد بود:

```
<Window x:Class="WpfRptTests.MainWindow"
      xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
      xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
      xmlns:c="clr-namespace:CodeReason.Reports.Controls;assembly=CodeReason.Reports"
      xmlns:vm="clr-namespace:WpfRptTests.ViewModel"
      Title="MainWindow" WindowState="Maximized" Height="350" Width="525">
  <Window.Resources>
      <vm:ProductViewModel x:Key="vmProductViewModel" />
  </Window.Resources>
  <Grid DataContext="{Binding Source={StaticResource vmProductViewModel}}">
      <c:BusyDecorator IsBusyIndicatorHidden="{Binding RptGuiModel.IsBusyIndicatorHidden}">
          <DocumentViewer Document="{Binding RptGuiModel.Document}"  />
      </c:BusyDecorator>
  </Grid>
</Window>
```

تعریف ابتدایی RptGuiModel به صورت زیر است (جهت مشخص سازی مقادیر IsBusyIndicatorHidden و Document در حین بایندینگ اطلاعات):

```
using System.ComponentModel;
using System.Windows.Documents;

namespace WpfRptTests.Model
```

```
{
  public class RptGuiModel
  {
      public IDocumentPaginatorSource Document { get; set; }
      public bool IsBusyIndicatorHidden { get; set; }
  }
}
```

و این View اطلاعات خود را از ViewModel زیر دریافت خواهد نمود:

```
using System;
using System.Collections.Generic;
using System.IO;
using System.Threading;
using CodeReason.Reports;
using WpfRptTests.Helper;
using WpfRptTests.Model;

namespace WpfRptTests.ViewModel
{
  public class ProductViewModel
  {
      #region Constructors (1)

      public ProductViewModel()
      {
          RptGuiModel = new RptGuiModel();
          if (Stat.IsInDesignMode) return;
          //انجام عملیات نمایش گزارش در یک ترد دیگر جهت قفل نشدن ترد اصلی برنامه
          showReportAsync();
      }

      #endregion Constructors

      #region Properties (1)

      public RptGuiModel RptGuiModel { set; get; }

      #endregion Properties

      #region Methods (3)

      // Private Methods (3)

      private static List<Product> getProducts()
      {
          var products = new List<Product>();
          for (var i = 0; i < 100; i++)
              products.Add(new Product { Name = string.Format("Product{0}", i), Price = i });

          return products;
      }

      private void showReport()
      {
          try
          {
              //Show BusyIndicator
              RptGuiModel.IsBusyIndicatorHidden = false;

              var reportDocument =
                  new ReportDocument
                  {
                      XamlData = File.ReadAllText(@"Report\SimpleReport.xaml"),
                      XamlImagePath = Path.Combine(Environment.CurrentDirectory, @"Report\")
                  };

              var data = new ReportData();

              // تعریف متغیرهای دلخواه و مقدار دهی آن‌ها
              data.ReportDocumentValues.Add("PrintDate", DateTime.Now);
              data.ReportDocumentValues.Add("RptBy", "وحید");

              // استفاده از یک سری اطلاعات آزمایشی به عنوان منبع داده
              data.DataTables.Add(getProducts().ToDataTable());

              var xps = reportDocument.CreateXpsDocument(data);
              //انقیاد آن به صورت غیر همزمان در ترد اصلی برنامه
```

```
                DispatcherHelper.DispatchAction(
                    () => RptGuiModel.Document = xps.GetFixedDocumentSequence()
                    );
            }
            catch (Exception ex)
            {
                //وجود این مورد ضروری است زیرا بروز استثناء در یک ترد به معنای خاتمه آنی برنامه است
                //todo: log errors
            }
            finally
            {
                //Hide BusyIndicator
                RptGuiModel.IsBusyIndicatorHidden = true;
            }
        }

        private void showReportAsync()
        {
            var thread = new Thread(showReport);
            thread.SetApartmentState(ApartmentState.STA); //for DocumentViewer
            thread.Start();
        }

        #endregion Methods
    }
}
```

توضیحات:
برای اینکه حین نمایش گزارش، ترد اصلی برنامه قفل نشود، از ترد استفاده شد و استفاده ترد به همراه DocumentViewer کمی نکته دار است:
- ترد تعریف شده باید از نوع STA باشد که در متد showReportAsync مشخص شده است.
- حین بایندیگ Document تولید شده توسط کتابخانه‌ی گزارشگیری به خاصیت Document کنترل، حتما باید کل عملیات در ترد اصلی برنامه صورت گیرد که سورس کلاس DispatcherHelper را در فایل پیوست خواهید یافت.

کل عملیات این ViewModel در متد showReport رخ می‌دهد، ابتدا فایل گزارش بارگذاری می‌شود، سپس متغیرهای سفارشی مورد نظر تعریف و مقدار دهی خواهند شد. در ادامه یک سری داده آزمایشی تولید و به DataTables گزارش ساز اضافه می‌شوند. در پایان XPS Document متناظر آن تولید شده و به کنترل نمایشی برنامه بایند خواهد شد.

دریافت سورس این مثال

نظرات خوانندگان

نویسنده: حسین مرادی نیا
تاریخ: ۱۳۸۹/۰۸/۱۹ ۲۳:۰۸:۰۴

سلام
خسته نباشید
من احساس می کنم بهتر باشه از همان ابزارهای تجاری استفاده کرد.چون اگه در آینده نیاز باشه برنامه و گزارش مربوطه پیشرفته تر بشته و یا توسعه داده بشه بهتر میشه این کار رو انجام داد.

نویسنده: وحید نصیری
تاریخ: ۱۳۸۹/۰۸/۲۰ ۰۰:۱۵:۳۴

این پروژه در حال حاضر دو ایراد داره:
- سر ستون‌های گزارش‌ها در صفحات مختلف تکرار نمی‌شوند.
- امکان گروه بندی ندارد.

مشکل حاد دیگری من ندیدم.

نویسنده: وحید نصیری
تاریخ: ۱۳۸۹/۰۸/۲۴ ۱۳:۳۲:۰۲

اگر علاقمند به تهیه خروجی از این گزارش‌ها باشید، کلاس‌های تهیه خروجی PDF/PNG/XPS را اینجا آپلود کردم:
(+)

همچنین اگر مایل باشید که به کنترل DocumentViewer دکمه‌های دلخواه تهیه خروجی را اضافه کنید می‌توان از قالب زیر کمک گرفت:
(+)