

در مطلب «[تولید پویای ستون‌ها در PdfReport](#)» عنوان شد که ذکر قسمت MainTableColumns و تمام تعاریف مرتبط با آن در PdfReports اختیاری است. همچنین به کمک متد MainTableAdHocColumnsConventions می‌توان بر اساس نوع‌های داده‌ای، بر روی نحوه نمایش ستون‌ها تاثیر گذاشت. برای مثال هرجایی DateTime مشاهده شد، به صورت خودکار تبدیل به تاریخ شمسی شود.

روش دیگری که این روزها در اکثر فریم‌های دات نتی مرسوم شده است، استفاده از Data Annotations جهت انتساب یک سری متادیتا به خاصیت‌های تعریف شده کلاس‌ها است. برای مثال ASP.NET MVC از این قابلیت زیاد استفاده می‌کند (در تولید پویای کد، یا اعتبار سنجی‌های سمت سرور و کاربر).

به همین جهت برای سازگاری بیشتر PdfReport با مدل‌های اینگونه فریم ورک‌ها، اکثر ویژگی‌ها و Data Annotations متداول را نیز می‌توان در PdfReport بکار برد. همچنین تعدادی ویژگی سفارشی نیز تعریف شده است، که در ادامه به بررسی آن‌ها خواهیم پرداخت.

آشنایی با مدل‌های بکار رفته در مثال جاری:

```
using System.ComponentModel;

namespace PdfReportSamples.Models
{
    public enum JobTitle
    {
        [Description("Grunt")]
        Grunt,

        [Description("Programmer")]
        Programmer,

        [Description("Analyst Programmer")]
        AnalystProgrammer,

        [Description("Project Manager")]
        ProjectManager,

        [Description("Chief Information Officer")]
        ChiefInformationOfficer,
    }
}
```

در اینجا یک enum، جهت تعیین سمت شغلی تعریف شده است. برای اینکه بتوان خروجی مطلوبی را در گزارشات شاهد بود، می‌توان از ویژگی Description، جهت تعیین مقدار نمایشی آن‌ها نیز استفاده کرد و این تعاریف در PdfReport خوانده و اعمال می‌شوند.

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.ComponentModel.DataAnnotations;
using PdfReportSamples.Models;
using PdfRpt.Aggregates.Numbers;
using PdfRpt.ColumnsItemsTemplates;
using PdfRpt.Core.Contracts;
using PdfRpt.Core.Helper;
using PdfRpt.DataAnnotations;

namespace PdfReportSamples.DataAnnotations
{
    public class Person
    {
        [IsVisible(false)]
        public int Id { get; set; }

        [DisplayName("User name")]
        //Note: If you don't specify the ColumnItemsTemplate, a new TextBlockField() will be used
    }
}
```

```

automatically.
[ColumnItemsTemplate(typeof(TextBlockField))]
public string Name { get; set; }

[DisplayName("Job title")]
public JobTitle JobTitle { set; get; }

[DisplayName("Date of birth")]
[DisplayFormat(DataFormatString = "{0:MM/dd/yyyy}")]
public DateTime DateOfBirth { get; set; }

[DisplayName("Date of death")]
[DisplayFormat(NullDisplayText = "-", DataFormatString = "{0:MM/dd/yyyy}")]
public DateTime? DateOfDeath { get; set; }

[DisplayFormat(DataFormatString = "{0:n0}")]
[CustomAggregateFunction(typeof(Sum))]
public int Salary { get; set; }

[IsCalculatedField(true)]
[DisplayName("Calculated Field")]
[DisplayFormat(DataFormatString = "{0:n0}")]
[AggregateFunction(AggregateFunction.Sum)]
public string CalculatedField { get; set; }

[CalculatedFieldFormula("CalculatedField")]
public static Func<IList<CellData>, object> CalculatedFieldFormula =
    list =>
    {
        if (list == null) return string.Empty;
        var salary = (int)list.GetValueOf<Person>(x =>
            x.Salary);

        return salary * 0.8;
    }; //Note: It's a static field, not a property.
}
}

```

مدل فوق جهت مقدار دهی اطلاعات یک شخص تعریف شده است.

- اگر قصد ندارید خاصیتی در این بین، در گزارشات ظاهر شود، از ویژگی `IsVisible` با مقدار `false` استفاده کنید.

- از ویژگی `DisplayName` جهت تعیین برچسب‌های سرستون‌ها استفاده خواهد شد.

- ذکر ویژگی `ColumnItemsTemplate` اختیاری است و اگر عنوان نشود به صورت خودکار از `TextBlockField` استفاده خواهد شد. اما اگر نیاز به استفاده از قالب‌های ستون‌های سفارشی و یا حتی قالب‌های پیش فرض دیگری که متنی نیستند، وجود دارد، می‌توانید از ویژگی `ColumnItemsTemplate` به همراه نوع کلاس مورد نظر استفاده نمایید. کلاس‌های پیش فرض قالب‌های ستون‌ها در `PdfReport` در پوشه `Lib\ColumnsItemsTemplates` سورس آن قرار دارند.

- برای تعیین نحوه فرمت اطلاعات در اینجا می‌توان از ویژگی `DisplayFormat` استفاده کرد. این ویژگی در اسمبلی `System.ComponentModel.DataAnnotations.dll` دات نت تعریف شده است؛ که در اینجا نمونه‌ای از استفاده از آن را برای تعیین نحوه نمایش تاریخ، ملاحظه می‌کنید. توسط این ویژگی حتی می‌توان مشخص ساخت (توسط پارامتر `NullDisplayText`) که اگر اطلاعاتی `null` بود، بجای آن چه عبارتی نمایش داده شود.

- اگر علاقمند به اعمال تابعی تجمعی به ستونی خاص هستید، از ویژگی `CustomAggregateFunction` استفاده کنید. پارامتر آن نوع کلاس تابع مورد نظر است. یک سری تابع تجمعی پیش فرض در فضای نام `PdfRpt.Aggregates.Numbers` قرار دارند. البته امکان تهیه انواع سفارشی آن‌ها نیز پیش بینی شده است که در قسمت‌های بعد به آن خواهیم پرداخت.

- امکان تعریف خواص محاسباتی نیز پیش بینی شده است. برای این منظور دو کار را باید انجام داد:

الف) ویژگی `IsCalculatedField` را با مقدار `true` بر روی خاصیت مورد نظر اعمال کنید.

ب) هم نام خاصیت محاسباتی افزوده شده به کلاس جاری، ویژگی `CalculatedFieldFormula` را بر روی یک فیلد استاتیک عمومی در آن کلاس به نحوی که ملاحظه می‌کنید (مطابق امضای فیلد `CalculatedFieldFormula` فوق)، تعریف نمایید. (علت این است که نمی‌توان توسط ویژگی‌ها از `delegates` استفاده کرد و این محدودیت ذاتی وجود دارد)

در ادامه کدهای منبع داده فرضی مثال جاری ذکر شده است:

```

using System;
using System.Collections.Generic;
using PdfReportSamples.Models;

```

```

namespace PdfReportSamples.DataAnnotations
{
    public static class PersonnelDataSource
    {
        public static IList<Person> CreatePersonnelList()
        {
            return new List<Person>
            {
                new Person
                {
                    Id = 1,
                    Name = "Edward",
                    DateOfBirth = new DateTime(1900, 1, 1),
                    DateOfDeath = new DateTime(1990, 10, 15),
                    JobTitle = JobTitle.ChiefInformationOfficer,
                    Salary = 5000
                },
                new Person
                {
                    Id = 2,
                    Name = "Margaret",
                    DateOfBirth = new DateTime(1950, 2, 9),
                    DateOfDeath = null,
                    JobTitle = JobTitle.AnalystProgrammer,
                    Salary = 4000
                },
                new Person
                {
                    Id = 3,
                    Name = "Grant",
                    DateOfBirth = new DateTime(1975, 6, 13),
                    DateOfDeath = null,
                    JobTitle = JobTitle.Programmer,
                    Salary = 3500
                }
            };
        }
    }
}

```

در پایان، نحوه استفاده از منبع داده فوق جهت تامین یک گزارش، به نحو زیر می‌باشد:

```

using System;
using PdfRpt.Core.Contracts;
using PdfRpt.FluentInterface;

namespace PdfReportSamples.DataAnnotations
{
    public class DataAnnotationsPdfReport
    {
        public IPdfReportData CreatePdfReport()
        {
            return new PdfReport().DocumentPreferences(doc =>
            {
                doc.RunDirection(PdfRunDirection.LeftToRight);
                doc.Orientation(PageOrientation.Portrait);
                doc.PageSize(PdfPageSize.A4);
                doc.DocumentMetadata(new DocumentMetadata { Author = "Vahid", Application = "PdfRpt",
Keywords = "Test", Subject = "Test Rpt", Title = "Test" });
            })
            .DefaultFonts(fonts =>
            {
                fonts.Path(Environment.GetEnvironmentVariable("SystemRoot") + "\\fonts\\tahoma.ttf",
Environment.GetEnvironmentVariable("SystemRoot") +
"\\fonts\\verdana.ttf");
            })
            .PagesFooter(footer =>
            {
                footer.DefaultFooter(printDate: DateTime.Now.ToString("MM/dd/yyyy"));
            })
            .PagesHeader(header =>
            {
                header.DefaultHeader(defaultHeader =>
                {
                    defaultHeader.ImagePath(AppPath.ApplicationPath + "\\Images\\01.png");
                    defaultHeader.Message("new rpt.");
                    defaultHeader.RunDirection(PdfRunDirection.LeftToRight);
                });
            });
        }
    }
}


```

```

    })
    .MainTableTemplate(template =>
    {
        template.BasicTemplate(BasicTemplate.ClassicTemplate);
    })
    .MainTablePreferences(table =>
    {
        table.ColumnsWidthsType(TableColumnWidthType.FitToContent);
    })
    .MainTableDataSource(dataSource =>
    {
        dataSource.StronglyTypedList(PersonnelDataSource.CreatePersonnelList());
    })
    .MainTableEvents(events =>
    {
        events.DataSourceIsEmpty(message: "There is no data available to display.");
    })
    .MainTableSummarySettings(summary =>
    {
        summary.OverallSummarySettings("Total");
        summary.PageSummarySettings("Page Summary");
        summary.PreviousPageSummarySettings("Previous Page Summary");
    })
    .MainTableAdHocColumnsConventions(adHocColumns =>
    {
        adHocColumns.ShowRowNumberColumn(true);
        adHocColumns.RowNumberColumnCaption("#");
    })
    .Export(export =>
    {
        export.ToExcel();
        export.ToXml();
    })
    .Generate(data => data.AsPdfFile(AppPath.ApplicationPath +
    "\\Pdf\\DataAnnotationsSampleRpt.pdf"));
    }
}

```

همانطور که مشخص است، از ذکر متد `MainTableColumns` به علت استفاده از `DataAnnotations` صرفنظر شده و `PdfReport` این تعاریف را بر اساس ویژگی‌های خواص کلاس شخص دریافت می‌کند. تنها از متد `MainTableAdHocColumnsConventions` جهت مشخص سازی اینکه نیاز به نمایش ستون ردیف می‌باشد، استفاده کرده‌ایم.

 new rpt.						
#	User name	Job title	Date of birth	Date of death	Salary	Calculated Field
1	Edward	Chief Information Officer	01/01/1900	10/15/1990	5,000	4,000
2	Margaret	Analyst Programmer	02/09/1950	-	4,000	3,200
3	Grant	Programmer	06/13/1975	-	3,500	2,800
Page Summary					12,500	10,000
Total					12,500	10,000