



# Fast-Report open source

## หัวข้ออธิบาย

- Fast-Report คืออะไร จะมาช่วยเราทำงานอย่างไร
- แนวทางการนำไปใช้หรือ Implementations
- Fast-Report designer เครื่องมือที่เจ๋งและใช้งานง่าย สำหรับ Fast-Report
- แล้วเสร็จจะพากำ Workshop ง่ายๆ เป็นตัวอย่างซัก 1-2 ตัวอย่าง

## Fast-Report คืออะไร

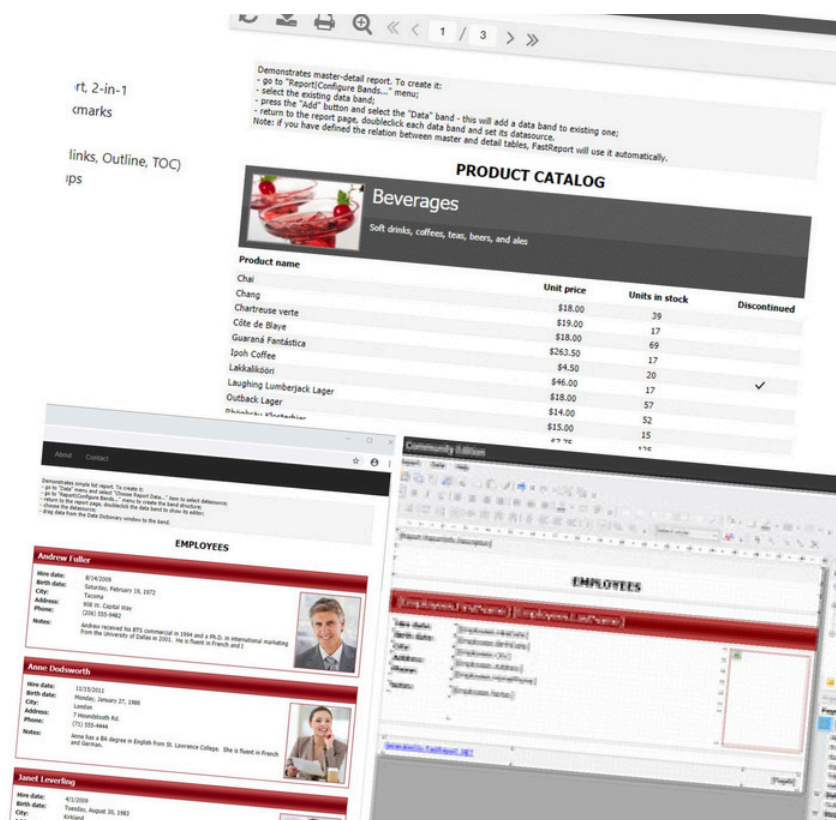
Tools ที่จะมาช่วยเราในการออกรายงานได้ ฟรี ง่าย และรวดเร็วขึ้น

Open source report generator สำหรับ

- .NET 6
- .NET Core
- .NET Framework.

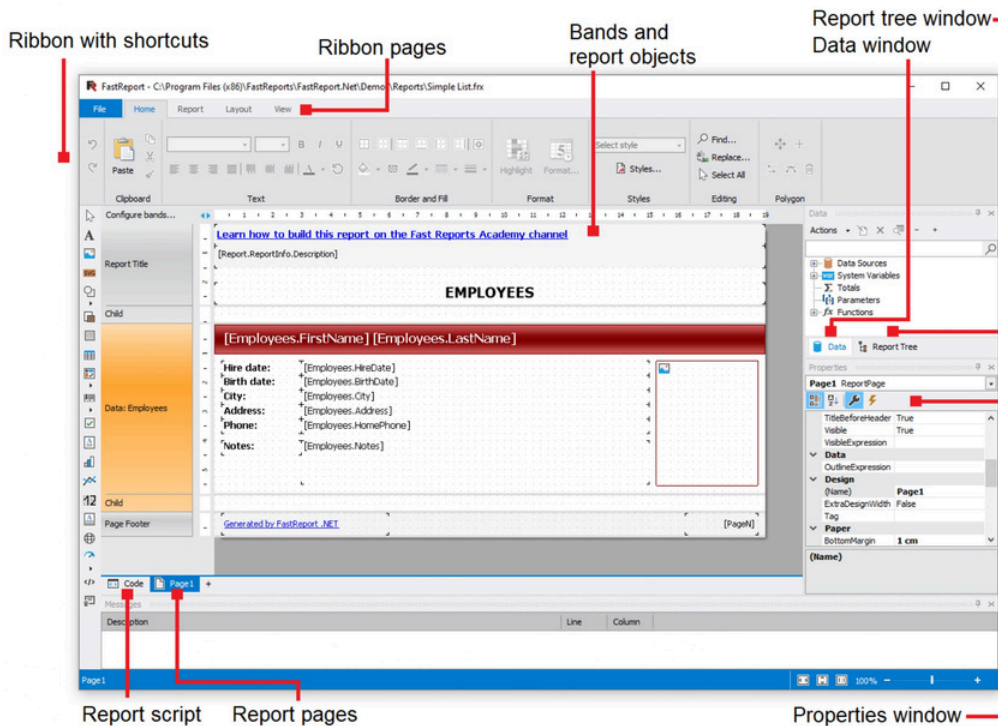
สามารถใช้ได้กับ

- MVC
- Web API
- Console Applications



## Fast-Report คืออะไร

Fast report ถ้าจะให้พูดง่ายๆคือ fast report เรามีแผนจะใช้ตัว fast-report เข้ามาช่วยในการจะการไฟล์ report ที่เป็นตัว PDF โจทย์ก็คือใช้งานง่าย (ตอบโจทย์) รวดเร็ว (ตอบโจทย์) learning cruve ต่ำ เนื่องจากการ Implement ไม่ยากและสามารถ design report ได้หลากหลาย มาพร้อมกับ designer ส่วนตัวอีกต่างหากนะครับ ของดีจัดเลย สามารถใช้งานร่วมกับ .NET6/.NET Core/.NET Frameworks ที่เป็นทั้ง API, MVC รวมไปถึง console application เลยนะครับทุกคน ที่สำคัญคือมันมี version free ครับ และที่ผมจะหยิบมาพูดคุยในวันนี้ก็คือ ตัวฟรี Implement เป็น Web API ให้ทุกๆคนที่เข้ามาฟังได้ดูกันครับ ..



# Report Designer

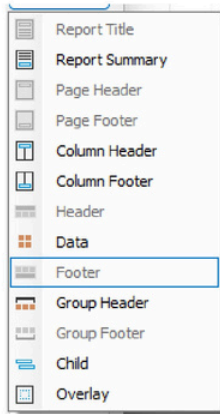
และไม่เป็นการเสียเวลาทุกคนไปมากกว่านี้และไม่ให้ผมออกทะเลไปมากกว่านี้เรามาเข้าเรื่องกันดีกว่านะครับ

ตัว Fast-Report ที่ผมจะยกมาพูดในวันนี้มันเป็น version free ก็จริงแต่ก็ต้องแลกมาด้วยข้อจำกัดนิดหน่อย และผมคิดว่าตัวมันเองสามารถตอบโจทย์ ได้ดีกว่าหลายๆ tools แน่แน่นอนครับโดย โดยภาพในสไลด์นี้จะเป็นตัวอย่าง designer ที่เราจะมาใช้ทำตัว template ให้กับ report ของเรานั้นเองครับขอไปอย่างไวๆ ละกันครับผมเชื่อว่าทุกคนจะคุ้นชินเอง และจะจำได้ดีกว่าเมื่อเห็นตัวอย่างการใช้งานของจริงๆ

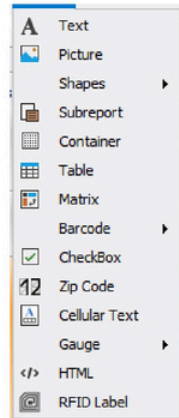
- Ribbon with shortcuts and Ribbon pages
  - จะรวมเครื่องมือที่ใช้จัดการกับสิ่งต่างๆตาม page ไว้เช่น Page File ก็จะมีเครื่องมือจัดการเกี่ยวกับ file ไม่ว่าจะเป็น import, open export เป็นต้น
- Bands and report objects
- Report tree window
  - จะแสดงรายละเอียดเกี่ยวกับ object ภายใต bands นั้นๆ
- Data window
  - นำเข้าส่งออก พร้อมกับ build in function ต่างๆ ก็จะอยู่ในนี้
- Properties window
  - คุณสมบัติของ bands และ objects
- Report pages
  - จัดการหน้า จัดลำดับก่อนหลัง เพิ่ม ลบ เลือก
- Report script
  - ส่วนนี้เอาไว้เขียนการทำงานที่ซับซ้อน แต่ logic ตรงนี้จะมี usecase ให้ใช้ไม่เยอะผมมองว่าไม่ควรนำไปใช้เกี่ยวกับการ render ตัว data แต่สามารถใช้เป็นเสริมในส่วนอื่นๆ ได้ยกตัวอย่างเช่นการนับ page เป็นต้น

# Open Source Designer

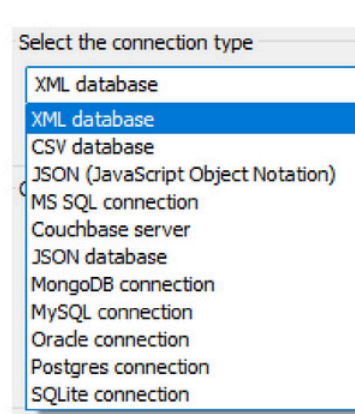
Band-oriented: 13 types



Objects: 8 types



DataSource: 11 types



Export: 7 types

- HTML
- BMP
- PNG
- JPEG
- GIF
- TIFF
- EMF

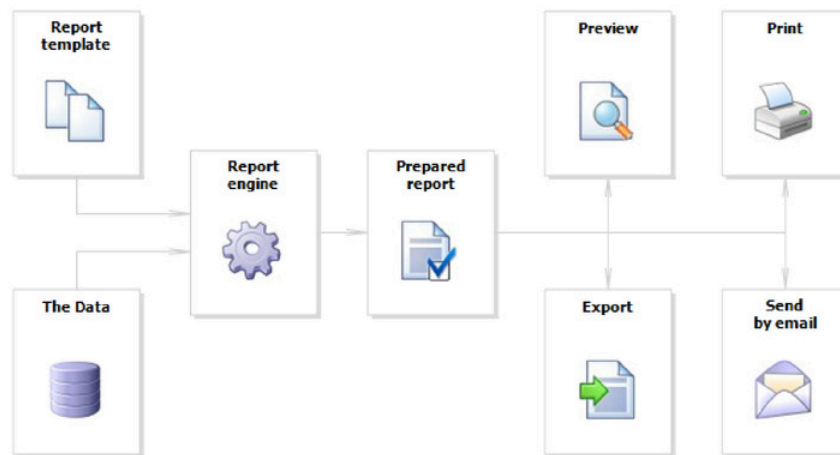
+ BusinessObjectDataSource

- FastReport has ability to get data from business objects of IEnumerable type.

## ตัว Open source designer มี

- Bands ให้เราได้ใช้ถึง 13 bands กันเลยนะครับทุกคน ไม่ว่าจะเป็น Title, Summary, Header, Footer ของ page กับ column และ group หรือแยกออกมาชัดเจนซึ่งให้ header กับ footer มาเยอะมากจริงๆนะครับต้องขอบอกไว้ตัวอย่างที่ผมเตรียมมาวันนี้ก็ใช้ไม่หมดทุกอันเหมือนกันนะครับ และ data ตัวหลักของเรานะครับ และปิดท้ายด้วย child and overlay
- Objects มา 8 types ไม่ว่าจะเป็น text, picture, shapes, sub report, containers, table, matrix, barcode, checkbox, zip code, calculator, gauge, html, RFID label
- DataSource: 11 types XML, CSV, JSON, MSSQL, Couchbase, MongoDB, MySQL, Oracle, Postgres, SQLite
- Export 7 types: HTML, BMP, PNG, JPEG, GIF, TIFF, EMF และจริงแล้วมันยังสามารถทำ Preview และ PDF ได้อีกด้วยนะไม่ธรรมดาแต่ต้องติดตั้ง package เพิ่มเติมนิดหน่อยเดี๋ยวจะพูดถึงทีหลังนะครับ

# Implementations



และก็มาถึงหัวข้อที่ทำคนรอคอยนะครับ Implementations มาถึงตรงนี้หลายๆคนอาจจะสงสัยแล้วแหละว่าเออพูดมาซะเยอะแล้วสรุปมันเอาไปใช้ยังไงกันหละ ซึ่งไม่ขอพูดพ้าทำเพลงนะครับเข้าเรื่องกันดีกว่า

1. เราต้องมีสิ่งตั้งต้นให้มันอยู่ 2 อย่างนะครับ หนึ่งคือ Template ครับซึ่งจะมาในรูปแบบของ file นามสกุล .frx ซึ่งมีโครงสร้างด้านในเป็น XML นั่นเองครับ กับสอง Data ครับ จะปรากฏใน code และจะต้องเขียนเป็น List หรือ IEnumerable เสมอนะครับ
2. จาแผนภาพมันจะทำ 2 อย่างนี้ไป cook ใน Report engine ครับเทียบกับการทำอาหารขั้นตอนนี้จะเป็นการเตรียมวัตถุดิบครับมีได้ก็เอาไปต้มหมักไปดับคาวเปป็นต้น
3. พอเสร็จครับก็จะมาถึงขั้นที่ชื่อว่า Prepared report เสร็จจากการเตรียมของแล้วพร้อมจะเป็นพัสดุเพราะแล้วครับ
4. และ Report ที่เตรียมเสร็จเราก็จะนำมาทำได้ 4 แบบหลักๆ ด้วยกันครับ Preview, Export, Print, Send by Email แต่ตัวอย่างวันนี้จะคุยแค่ Preview, กับ Export ครับ 2 ตัวแบบเลย

## ReportTemplateI.ftx

```
<?xml version="1.0" encoding="utf-8"?>
<Report ScriptLanguage="CSharp" ReportInfo.Name="testName" ReportInfo.Author="testAuthor"
ReportInfo.Version="1.0.1" ReportInfo.Description="testDescription" ReportInfo.Created="04/03/2025
14:18:00" ReportInfo.Modified="04/04/2025 16:17:57" ReportInfo.CreatorVersion="2025.1.0.0">
  <DataSource>
    <DataSourceObject DataSourceName="JSDM"
      DataType="System.Collections.Generic.List`1[SimpleFastReport.API.DTOs.EmployeeResponseDTO,
SimpleFastReport.API, Version=1.0.0.0, Culture=neutral, PublicKeyToken=null]" Enable="true">
      <Column Name="EmployeeId" DataType="System.Double"/>
      <Column Name="FirstName" DataType="System.String"/>
      <Column Name="LastName" DataType="System.String"/>
    </DataSourceObject>
  </DataSource>
  <ReportPage Name="Page0" Watermark.Font="Arial, 60pt">
    <OverlayBand Name="Overlay1" Width="718.2" Height="810.55" XCoord="387.45,387.45">
      <TextObject Name="Text12" Width="718.2" Height="810.55" Content="true" Scale="true"
        CanResize="false" Font="UpperCase" Employees List" Padding="2, 2, 2, 2" AutoResize="true"
        Horizontal="Center" Vertical="Center" Font="Leelawadee, 36pt"/>
    </OverlayBand>
    <ReportPage Name="Page1" Watermark.Font="Arial, 60pt" ResetPageNumber="true">
      <ReportTitleBand Name="ReportTitle" Width="718.2" Height="37.8">
        <TextObject Name="Text1" Width="718.2" Height="37.8" Text="Upper Case" Employees List"
          Horizontal="Center" Vertical="Center" Font="Leelawadee, 36pt, style=bold"/>
      </ReportTitleBand>
      <TableBand Name="Table1" Top="41.8" Width="718.2" Height="18.9">
        </TableBand>
      <ReportTitleBand>
        <GroupHeaderBand Name="GroupHeader1" Top="64.7" Width="718.2" Height="37.8" Fill="LinearGradient"
          FillStartColor="LightCoral" FillEndColor="Maroon" FillAngle="270" FillFocus="8.5" FillContrast="1"
          Condition="12508.EmployeesList" Repeat="true">
          <TextObject Name="Text11" Left="18.9" Width="688.4" Height="37.8" Text="{12508.FirstName}
{12508.LastName}" Vertical="Center" Font="Leelawadee, 36pt, style=bold" TextFillColor="White"/>
        </GroupHeaderBand>
        <PageFooterBand Name="PageFooter1" Top="776.15" Width="718.2" Height="18.9">
          <TextObject Name="Text1" Left="114.25" Width="144.5" Height="18.9" Text="{TotalPages} / {TotalPages}"
            Font="Arial, 36pt"/>
        </PageFooterBand>
      </ReportPage>
    </Report>
```

+

## ReportDataI.json

```
{
  "employees": [
    {
      "employeeId": 1,
      "firstName": "John",
      "lastName": "Doe",
      "email": "john.doe@example.com",
      "phone": "123-456-7890",
      "hireDate": "2022-06-15T00:00:00",
      "salary": 55000,
      "department": "IT",
      "imageUrl": "https://lineagentapi.uat.siamail.com/messageImages/638425/tnape376369_78250483_095648.jpg"
    }
  ]
}
```

นี่ก็จะเป็นหน้าตาของสิ่งของ 2 อย่างที่เราจะเตรียมไปใส่ตัว Report engine นะครับ ...

## Report Engine

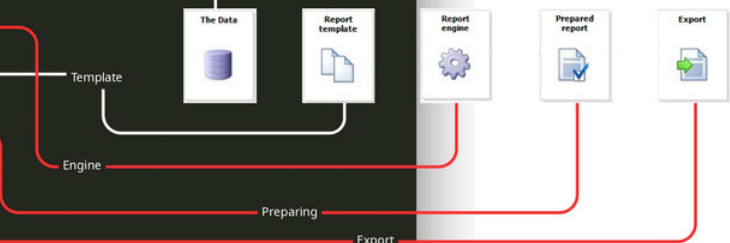
```
var data = await _dbContext.Products
.Include(_ => _.ProductGroup)
.Where(_ => _.IsActive == true)
.AsSplitQuery()
.ToListAsync();

var report = new Report();

try {
  report.Load("ReportTemplateI.ftx");
  report.RegisterData(data, "DataSourceName");
  report.Prepare();

  using (MemoryStream ms = new MemoryStream())
  {
    PDFSimpleExport pdf = new PDFSimpleExport();
    report.Export(pdf, ms);
    return File(ms.ToArray(), "application/pdf", "ExampleReport.pdf");
  }
} catch (Exception ex) {
  return BadRequest($"Error generating report: {ex.Message}");
} finally {
  report.Dispose();
}
```

```
<PackageReference Include="FastReport.OpenSource" Version="2025.1.0" />
<PackageReference Include="FastReport.OpenSource.Export.PdfSimple" Version="2025.1.0" />
<PackageReference Include="FastReport.OpenSource.Data.Json" Version="2021.4.0" />
<PackageReference Include="FastReport.OpenSource.Web" Version="2025.1.0" />
```



อะพอมาถึงหน้านี้ก็จะเห็นตัวอย่าง Process ต่างเป็น ภาพมากขึ้นครับ

- Data จะเป็นว่าจะเป็นการ query จาก EF-core เลยใช้ไหมหะครับซึ่งนั้นก็หมายความว่าตรงนี้ถ้า Data มีความซับซ้อนเกินไปใช้ Stored ได้นั้นเองครับ
- จะเห็นแต่ละขั้นตอนชัดเจนแต่เดี่ยวพอทำจริงจะมี File Helper ที่ถูกเขียนไว้แล้วจะเข้ามาช่วยอีกทีครับ

# Workshop

ครับ Workshop เดี่ยวผมจะทำตัวอย่างให้ดูว่าที่พูดมาทั้งหมดสุดท้ายแล้วเอาไปใช้จริงทำยังไง

1. Import ExampleFastReportData.bacpac database
2. Reverse engineer
3. Cooking

Source Code:

🌐 [GitHub - Pratchaya0/csharp-fast-report-example](#): This repo for stored using "Fast report" w...

## ข้อควรระวัง

- การทำ Report preview ในที่นี้เป็นแค่ตัวอย่างเบื้องต้นอาจจะเป็นปัญหาด้านความปลอดภัยได้ถ้าหากนำไปใช้จริงๆ
- การทำ Report ต้องจัดการกับเรื่อง DATA สำคัญ ไม่ควรมากเกินไป
- การเปลี่ยน FONT สามารถอ่านได้ใน Handout



## ข้อสังเกต

- File .frx ต้องเปลี่ยน Data source เป็น **BusinessObjectDataSource** + เพิ่ม **ReferenceName** ให้ด้วยครับ

```
<?xml version="1.0" encoding="utf-8"?>
<Report ScriptLanguage="CSharp" ReportInfo.Created="04/08/2025 13:13:22" ReportInfo.Modified="04/08/2025 14:35:11"
ReportInfo.CreatorVersion="2025.1.0.0">
  <Dictionary>
    <BusinessObjectDataSource Name="Employees" ReferenceName="Employees" DataType="System.Int32" Enabled="true">
      <Column Name="employeeId" DataType="System.Double"/>
      <Column Name="firstName" DataType="System.String"/>
    ...
```

- MVC
  - View จะทำงานได้ต้องเรียกใช้ service มันด้วย

```
services.AddMvc();
```



- การใช้ Font มี 2 แบบคือ
  - เรียกใช้ภายใน code ต้องแก้ Helper นิดหน่อยเพิ่ม Code ส่วนนี้ไปก่อน Prepare()

```
using System.Drawing.Text;
using System.Runtime.InteropServices;

// Path to your .ttf font file
var fontCollection = new PrivateFontCollection();
fontCollection.AddFontFile("path/to/your-font.ttf");

// Get the first font family in the collection
FontFamily customFontFamily = fontCollection.Families[0];

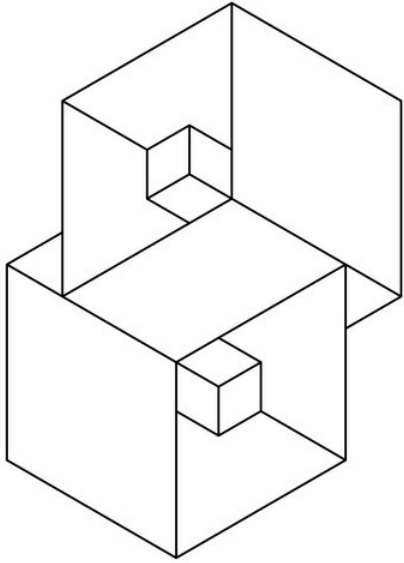
var customFont = new Font(customFontFamily, 10f, FontStyle.Regular);

foreach (ReportComponentBase page in report.Pages)
{
    foreach (var obj in page.AllObjects)
    {
        if (obj is FastReport.TextObject textObject)
        {
            textObject.Font = customFont;
        }
    }
}

var pdf = new PDFSimpleExport
{
    EmbeddingFonts = true
};

report.Export(pdf, ms);
```

- Install font ลงเครื่องแล้วเรียกผ่าน Fast-report designer



## Reference

Source Code: <https://github.com/Pratchaya0/csharp-fast-report.git>

Fast-Report Manual: [https://www.fast-report.com/public\\_download/docs/FRNet/online/en/index.html](https://www.fast-report.com/public_download/docs/FRNet/online/en/index.html)

Fast-Report Repository: <https://github.com/FastReports/FastReport>

Fast-Report OpenSource v2025.2.0: <https://github.com/FastReports/FastReport/releases/tag/v2025.2.0>

ก็ตรงนี้จะรวม link อ้างอิงไว้ นะครับในส่วนของ source code ถ้ามีเวลาที่จะหา usecase ตัวอย่างมาเติมให้เรื่อยๆ ครับ

ขอบคุณครับ



แล้วก็ขอบคุณที่ฟังมาถึงตรงนี้ ใครมีคำถามเพิ่มเติมสามารถสอบถามมาตรงนี้ได้เลยนะครับ หรือจะหลังไมค์ก็ได้เลยนะครับ

จ้ะก็ขอตัวลาไปก่อนถ้าผิดพลาดประการใดขออภัยมานะที่นี้ด้วยนะครับ ไว้เจอกันใหม่โอกาสหน้าครับ

ขอบคุณครับ