GENERATE BRADY REPORT SOLUTION

Contents

[Objective 1](#_Toc17006043)

[Stack 2](#_Toc17006044)

[Assumptions 3](#_Toc17006045)

[Sample Output 3](#_Toc17006046)

[References 5](#_Toc17006047)

[Please let me know what you think 5](#_Toc17006048)

# Objective

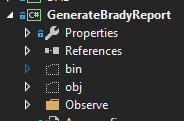
A folder is required to be observed for any incoming xml files of type - GenerationReport.xml

Input file contains different types of generators – Wind, Coal and Gas. Each of which have different fields

1. As soon as the file arrives in the observed folder, underlying parameters of output file of type – GenerationOutput.xml needs to be updated.
2. For the task, there is reference file containing the factors such as Value factor and emission factor is provided. Also, there is Generator- factor mapping provided in the requirement document which are a constant

Implementation

1. Input – Observe folder. For simplicity, this folder is added in solution folder itself.



1. Usage -On the **App.config** file shown above, key **observeDirectoryPath** is present, which can be used to update the input **observe** folder
2. When a file of type .XML is added, it is automatically picked and processed.
3. Process- The parameter fields are merged and accordingly the output xml is generated.
4. Same output file GenerationOutput.xml(value as per **OutputFilePath key in App.config**) is updated after the details are processed. Incoming files can be received any time and accordingly the parameter changes. Output report is cumulative in nature as files can be received any time.

# Stack

**Console application** - .NET Framework 4.6.1

**Libraries** – System.xml, System.xml.linq, System.Configuration, System.Collection.Generic

**Error Logging** – On to console application window.

# Assumptions

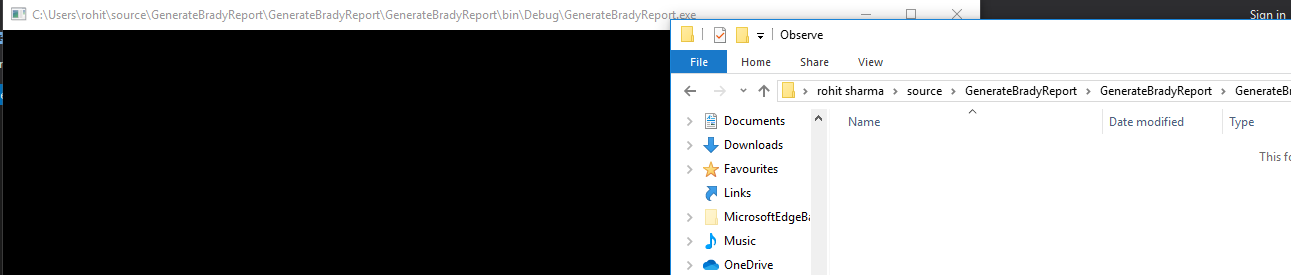
As new files can be reaching input folder, the process is cumulative and hence a single file output is assumed.

Calculations are as per below

1. **Total** **Generation Value** for each generator – For an existing output file, generator output value - total is added to existing total of the generator. So for ex – if there is a file at 10 am with Coal[1] as generator name and followed by this another file at 11 am with Coal[1] as generator name. Then the output field value – Total gets added to the existing total of Coal[1].
2. **Highest** **Daily Emissions -** For each of the incoming files, the emission value gets added to the existing emission for each of the generators and the maximum emission generator details is provided in the output for each day.
3. Output **Actual Heat Rate** for each coal generator. As this field should be cumulative, **TotalHeatInput** and **ActualNetGeneration** are assumed to be updated in each of the incoming files. And hence the values in the last file are considered for its calculation.
4. For implementation of step 3, *It is required to calculate and output Actual Heat Rate for each coal generator.* FieldActualHeatRate is implemented as a list to be contained inside ActualHeatRates as this can be a list of coal generators.

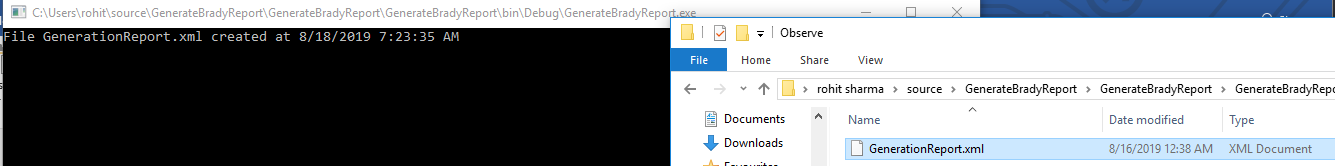
# Sample Output

Before



After –

After addition of a correct xml file(type GenerateReport.xml – with Wind,gas,coal data in same format) in observe folder, it is picked and processed.



Sample Output -

<?xml version="1.0" encoding="utf-8"?>

<GenerationOutput xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xsd="http://www.w3.org/2001/XMLSchema">

<totals>

<Generator>

<Name>Wind[Offshore]</Name>

<Total>24939.26</Total>

</Generator>

<Generator>

<Name>Wind[Onshore]</Name>

<Total>73041.8047</Total>

</Generator>

<Generator>

<Name>Gas[1]</Name>

<Total>127683.781</Total>

</Generator>

<Generator>

<Name>Coal[1]</Name>

<Total>80125.7344</Total>

</Generator>

</totals>

<maxEmissionGenerators>

<Day>

<Name>Coal[1]</Name>

<Date>2017-01-01T05:30:00+05:30</Date>

<Emission>159.812256</Emission>

</Day>

<Day>

<Name>Coal[1]</Name>

<Date>2017-01-02T05:30:00+05:30</Date>

<Emission>1591.279</Emission>

</Day>

<Day>

<Name>Gas[1]</Name>

<Date>2017-01-03T05:30:00+05:30</Date>

<Emission>46.99911</Emission>

</Day>

</maxEmissionGenerators>

<actualHeatRates>

<ActualHeatRate>

<Name>Coal[1]</Name>

<HeatRate>1</HeatRate>

</ActualHeatRate>

</actualHeatRates>

</GenerationOutput>

# References

* <https://stackoverflow.com/questions/8373552/serialize-an-object-to-xelement-and-deserialize-it-in-memory>
* <https://stackoverflow.com/questions/40859136/filesystemwatcher-with-the-console-application>

# Please let me know what you think

Thanks!