

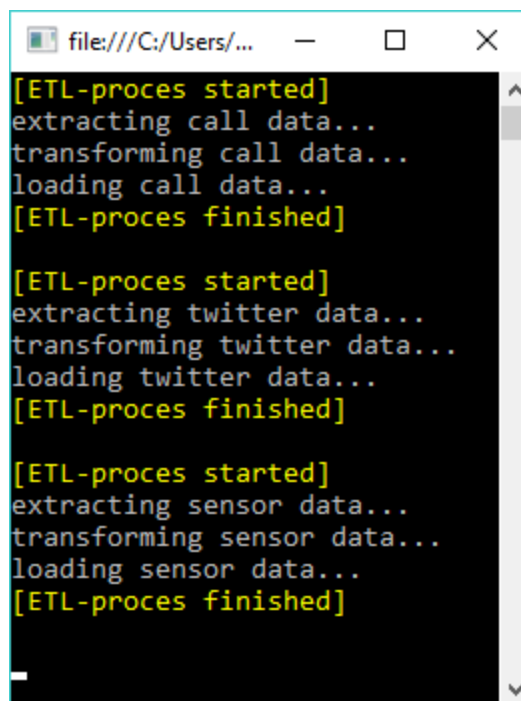
## Opdracht 1 ('Template pattern')

In de BigData-wereld gaat het inladen van grote hoeveelheden data in een DataWareHouse vaak via het 'Extract-Transform-Load' (ETL) principe, waarbij data in het juiste formaat beschikbaar komt. Extract is het lezen van data uit een source database, Transform is het omzetten van de ingelezen data naar het gewenste target formaat (eventueel wordt de data gemerged met andere data), en Load is het schrijven van de omgezette data naar de target database. Deze 3 stappen worden in vaste volgorde uitgevoerd, maar de implementatie van elke stap is afhankelijk van de in-te-lezen data.

Maak een class `BigDataLoader` met een ETL-methode dat gebaseerd is op de 'Template Methode' pattern; van deze class worden geen instanties aangemaakt. Implementeer 3 specifieke classes `CallDataLoader`, `TwitterDataLoader` en `SensorDataLoader` die de (specifieke) implementatie verzorgen van de 3 ETL-stappen.

Implementeer een (eenvoudige) class `BatchProcessor` die een lijst van `BigDataLoaders` kan verwerken; uiteraard moet deze `BatchProcessor`-class een methode krijgen om een `BigDataLoader` toe te voegen aan de lijst, en een methode om de gehele batch te processen.

Maak een hoofdprogramma (Console) waarin een `BatchProcessor` wordt aangemaakt, en dat onderstaand scherm/uitvoer als resultaat heeft:

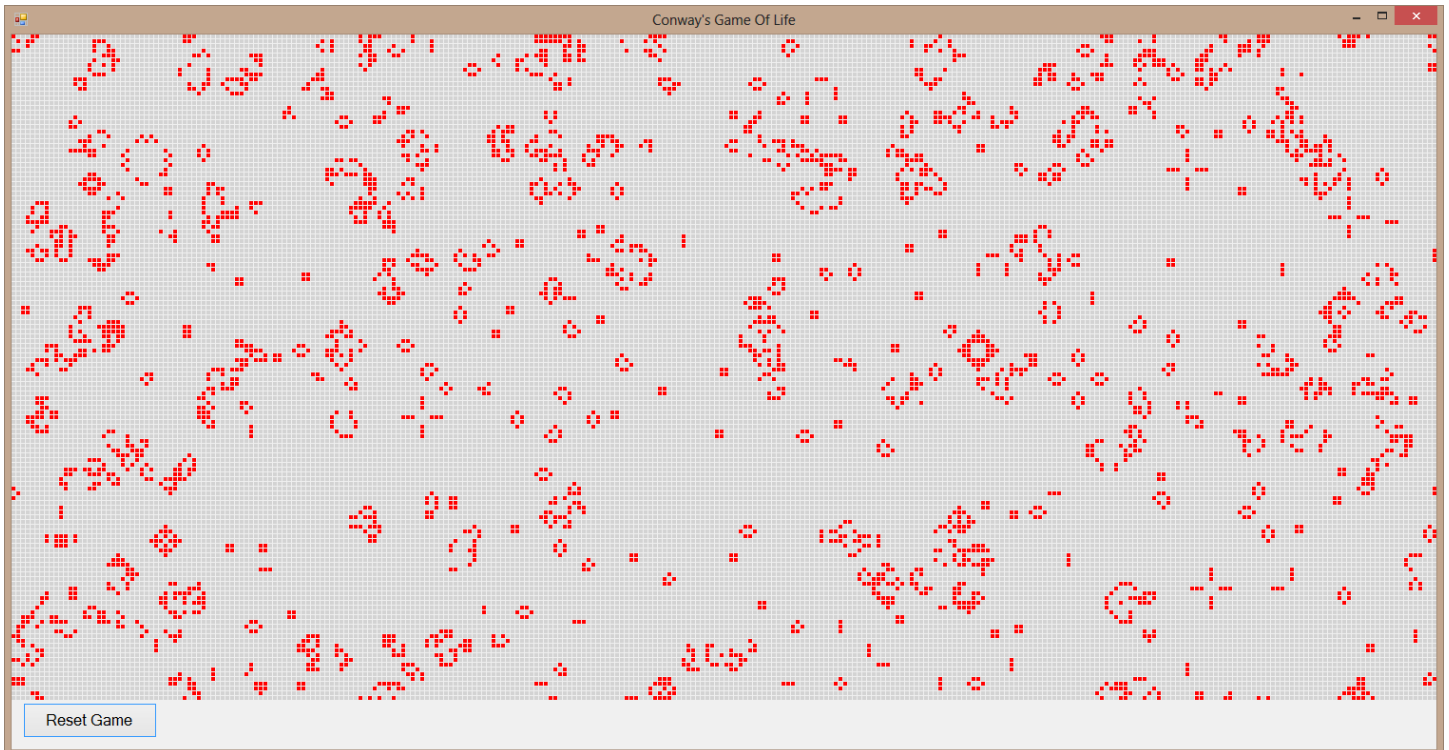


```
file:///C:/Users/...  
[ETL-proces started]  
extracting call data...  
transforming call data...  
loading call data...  
[ETL-proces finished]  
  
[ETL-proces started]  
extracting twitter data...  
transforming twitter data...  
loading twitter data...  
[ETL-proces finished]  
  
[ETL-proces started]  
extracting sensor data...  
transforming sensor data...  
loading sensor data...  
[ETL-proces finished]
```

## Opdracht 2 ('Template pattern')

Op Moodle vind je onder 'Les 2' applicatie 'Game of Life' (in een zip-file). Deze applicatie werkt volgens de standaard leefregels van John Conway (B3/S23), zie uitvoer in de screenshot hieronder. Wijzig deze applicatie zodat er 2 varianten mogelijk zijn: een 'standard life' en een 'high life' variant (B36/S23, zie <http://www.conwaylife.com/wiki/HighLife>). Doe dit door de Template-pattern toe te passen op class `ConwayGameOfLife`, waarbij methode 'CellShouldLive' door de 2 afgeleide classes (StandardLife en HighLife) geïmplementeerd moet worden. Deze methode 'CellShouldLive' wordt aangeroepen in methode 'Evolve'.

Voor meer informatie over Conway's Game of Life, zie o.a. [https://nl.wikipedia.org/wiki/Game\\_of\\_Life](https://nl.wikipedia.org/wiki/Game_of_Life).



Ga voor jezelf na welke methode de 'Template Method' is.

## Opdracht 3 ('Observer pattern')

Er moet een (Console) applicatie geschreven worden waarin een MP3-player nummers afspeelt. Verschillende displays tonen het huidige nummer van de MP3-player, elk op hun eigen manier.

De volgende classes moeten geïmplementeerd worden: 'MP3Player', 'SimpleMP3Display' en 'FancyMP3Display'.

De MP3Player class implementeert de IObservable interface:

```
public interface IObservable
{
    void AddObserver(IObserver observer);
    void RemoveObserver(IObserver observer);
    void NextSong();
}
```

De MP3Player class heeft o.a. de volgende properties/members/methoden:

```
public Song CurrentSong { get; private set; }
public void NextSong() { ... }
```

Methode 'NextSong' selecteert (random) een volgend nummer en informeert alle aangemelde observers van de wijziging. De verschillende nummers kun je in lijst plaatsen (`List<Song> songs`).

De 2 display classes (SimpleMP3Display en FancyMP3Display) implementeren de IObserver interface (zodat de MP3player ze kan updaten/informeren):

```
public interface IObserver
{
    void Update(Song song);
}
```

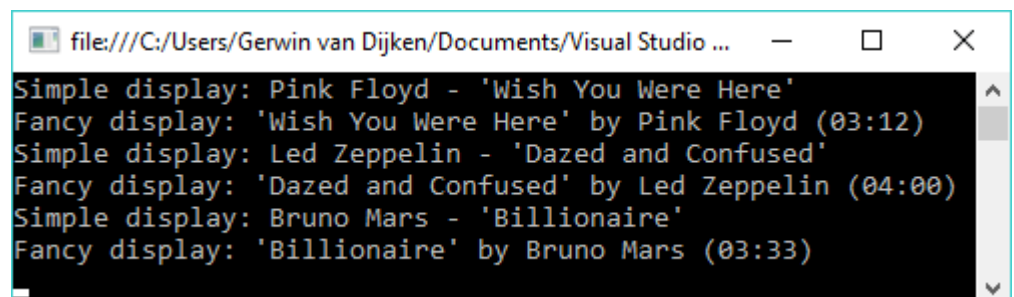
Hieronder vind je het hoofdprogramma en de te verwachten uitvoer:

```
class Program
{
    static void Main(string[] args)
    {
        // maak een MP3 player aan
        IObservable player = new MP3Player();

        // maak de displays aan
        IObserver mp3Display1 = new SimpleMP3Display(player);
        IObserver mp3Display2 = new FancyMP3Display(player);

        // zet player op een nieuw nummer
        // (aangezien er geen hardware is, doen we dat hier...)
        player.NextSong();
        player.NextSong();
        player.NextSong();

        Console.ReadKey();
    }
}
```



```
Simple display: Pink Floyd - 'Wish You Were Here'
Fancy display: 'Wish You Were Here' by Pink Floyd (03:12)
Simple display: Led Zeppelin - 'Dazed and Confused'
Fancy display: 'Dazed and Confused' by Led Zeppelin (04:00)
Simple display: Bruno Mars - 'Billionaire'
Fancy display: 'Billionaire' by Bruno Mars (03:33)
```