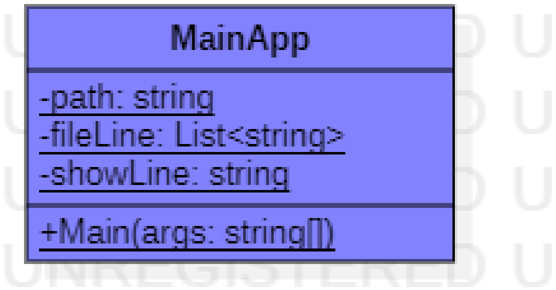
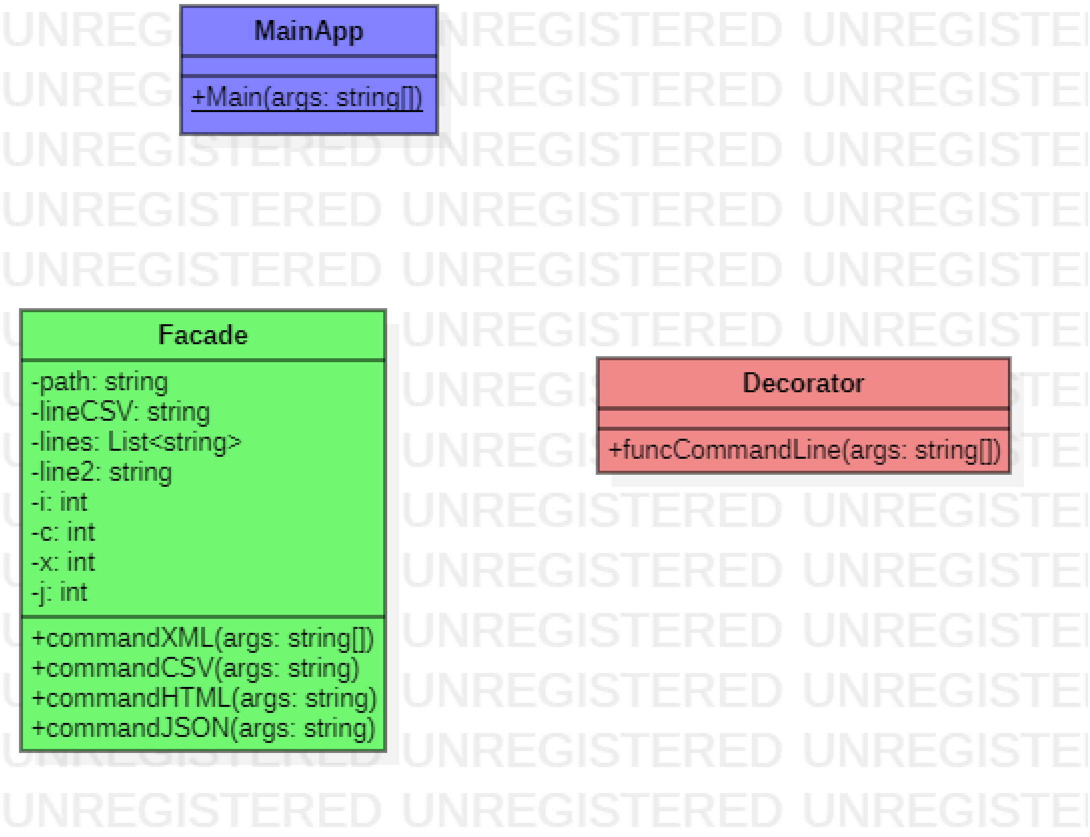
Programa Versión 1.0



```
using System;
using System.Collections.Generic;
using System.Data;
using System.IO;
using System.Linq;
using System.Runtime.Serialization.Formatters;
using System.Text;
using System.Threading.Tasks;

namespace V1
{
    class MainApp
    {
        static void Main(string[] args)
        {

        string path;
        string line2;
        List<string> lines;
```

```
            for (int i = 0; i < args.Length; i++)
            {
                path = @"..\..\..\" + args[i];
                lines = new List<string>();
                lines = File.ReadAllLines(path).ToList();
                foreach (String line in lines)
                {
                    line2 = line.Replace(' ', ',');
                    Console.WriteLine(line2);
                }
            }

        }
    }
}
```

Programa Versión 2.0


MainApp
+Main(args: string[])
Facade
-path: string
-lineCSV: string
-lines: List<string>
-line2: string
-i: int
-c: int
-x: int
-j: int
+commandXML(args: string[])
+commandCSV(args: string)
+commandHTML(args: string)
+commandJSON(args: string)
Decorator
+funcCommandLine(args: string[])

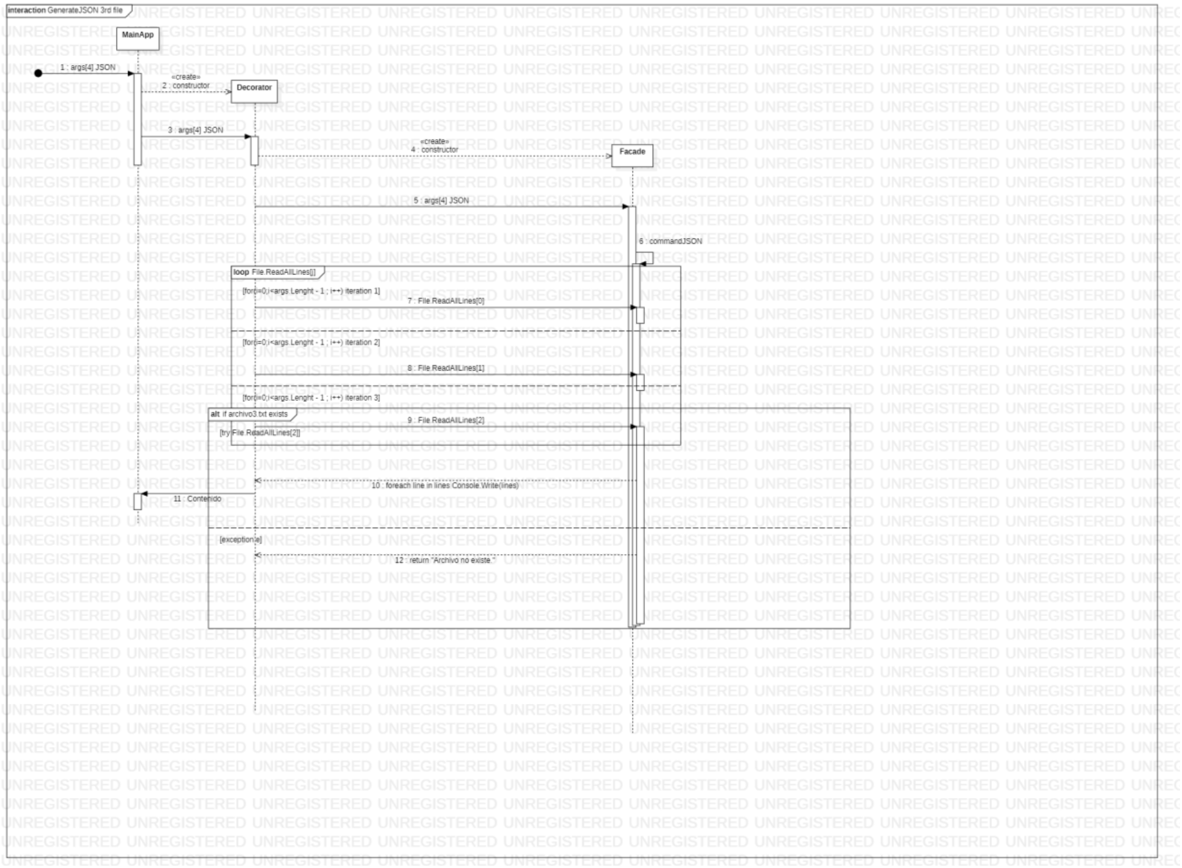
```
using System;
using System.Collections.Generic;
using System.Text;
using System.Linq;

namespace V2
{
    class Decorator
    {
        public void funcCommandLine(string[] args){
            Facade cmd = new Facade();

            if(args.Contains("HTML")){
                cmd.commandHTML(args);
            }else if(args.Contains("JSON")){
                cmd.commandJSON(args);
            }else if(args.Contains("CSV")){
                cmd.commandCSV(args);
            }else if(args.Contains("XML")){
                cmd.commandXML(args);
            }

        }
    }
}
```

```
using System;
using System.Collections.Generic;
using System.Text;
using System.IO;
using System.Linq;

namespace V2
{
    class Facade
    {
        private string path;
        private string lineCSV;
        private string line2;
        private List<string> lines;
        private int i, c, x, j;
        public void commandXML(string[] args)
        {

            for (x = 0; x < args.Length - 1; x++)
            {

                path = @"..\..\..\" + args[x];
                lines = new List<string>();
                lines = File.ReadAllLines(path).ToList();

                foreach (String line in lines)
                {
                    Console.WriteLine("<?xml version='1.0' enconding='UTF-16'?>");
                    Console.WriteLine("<project>");
                    Console.WriteLine("<Comment>");
                    Console.WriteLine("");

                    lineCSV = line.Replace('', ' ');
                    Console.WriteLine(lineCSV);

                    Console.WriteLine("");
                    Console.WriteLine("</Comment>");
                    Console.WriteLine("</project>");
                    Console.WriteLine("</xml");
                    Console.WriteLine("\n\n");
                }

            }

        }

        //------------------------------------------------------------------------
        public void commandCSV(string[] args)
        {



            for (c = 0; c < args.Length - 1; c++)
            {

                path = @"..\..\..\" + args[c];
                lines = new List<string>();
                lines = File.ReadAllLines(path).ToList();
                foreach (String line in lines)
                {
                    line2 = line.Replace(' ', ',');
                    Console.WriteLine(line2);
                    Console.WriteLine("\n\n");
                }

            }

        }

        //-----------------------------------------------------------------------------------------

        public void commandHTML(string[] args)
        {

            for (i = 0; i < args.Length - 1; i++)
            {

                path = @"..\..\..\" + args[i];
                lines = new List<string>();
                lines = File.ReadAllLines(path).ToList();

                foreach (String line in lines)
                {
                    Console.WriteLine("<DOCTYPE html>");
                    Console.WriteLine("<html lang='es'>");
                    Console.WriteLine("<head>");
                    Console.WriteLine("<title> </title>");
                    Console.WriteLine("</head>");
                    Console.WriteLine("<body>");
                    Console.WriteLine("<div>");
                    line2 = line.Replace(' ', ' ');
                    Console.WriteLine("<h5>" + line2 + "</h5>");
                    Console.WriteLine("</div>");
                    Console.WriteLine("</body>");
                    Console.WriteLine("</html>");
                    Console.WriteLine("\n\n");
                }

            }

        }

        //-----------------------------------------------------------------------------------------------



        public void commandJSON(string[] args)
        {

            for (j = 0; j < args.Length - 1; j++)
            {

                path = @"..\..\..\" + args[j];
                lines = new List<string>();
                lines = File.ReadAllLines(path).ToList();

                foreach (String line in lines)
```

```
                {
                    Console.WriteLine("\t[");
                    Console.WriteLine("\t\t{");
                    line2 = line.Replace(' ', ' ');
                    Console.WriteLine("\t\t\t'"+line2+"': 'string'");
                    Console.WriteLine("\t\t}");
                    Console.WriteLine("\t]");


                }


            }


        }






    }
}
```

```
using System;

namespace V2
{
    class MainApp
    {
        static void Main(string[] args)
        {
            Decorator f = new Decorator();
            f.funcCommandLine(args);
        }
    }
}
```

Secuence Diagram Generate_3RD_FILE_JSON


interaction GenerateJSON 3rd file
MainApp
Decorator
Facade
1 : args[4] JSON
«create»
2 : constructor
3 : args[4] JSON
«create»
4 : constructor
5 : args[4] JSON
6 : commandJSON
loop File.ReadAllLines[j]
[for(i=0;i<args.Lenght - 1 ; i++) iteration 1]
7 : File.ReadAllLines[0]
[for(i=0;i<args.Lenght - 1 ; i++) iteration 2]
8 : File.ReadAllLines[1]
[for(i=0;i<args.Lenght - 1 ; i++) iteration 3]
alt if archivo3.txt exists
9 : File.ReadAllLines[2]
[try File.ReadAllLines[2]]
10 : foreach line in lines Console.Write(lines)
11 : Contenido
[exception e]
12 : return "Archivo no existe."