



SUBDIRECCIÓN ACADÉMICA  
INSTITUTO TECNOLÓGICO DE TIJUANA TOMAS AQUINO  
DEPARTAMENTO DE SISTEMAS Y COMPUTACIÓN  
SEMESTRE:  
Agosto - Diciembre 2025

CARRERA:  
Ingeniería en Sistemas Computacionales

MATERIA:  
Patrones de Diseño de Software

Grupo:  
02:00 - 03:00 pm

Unidad:  
IV

Actividad:  
Examen Unidad 4

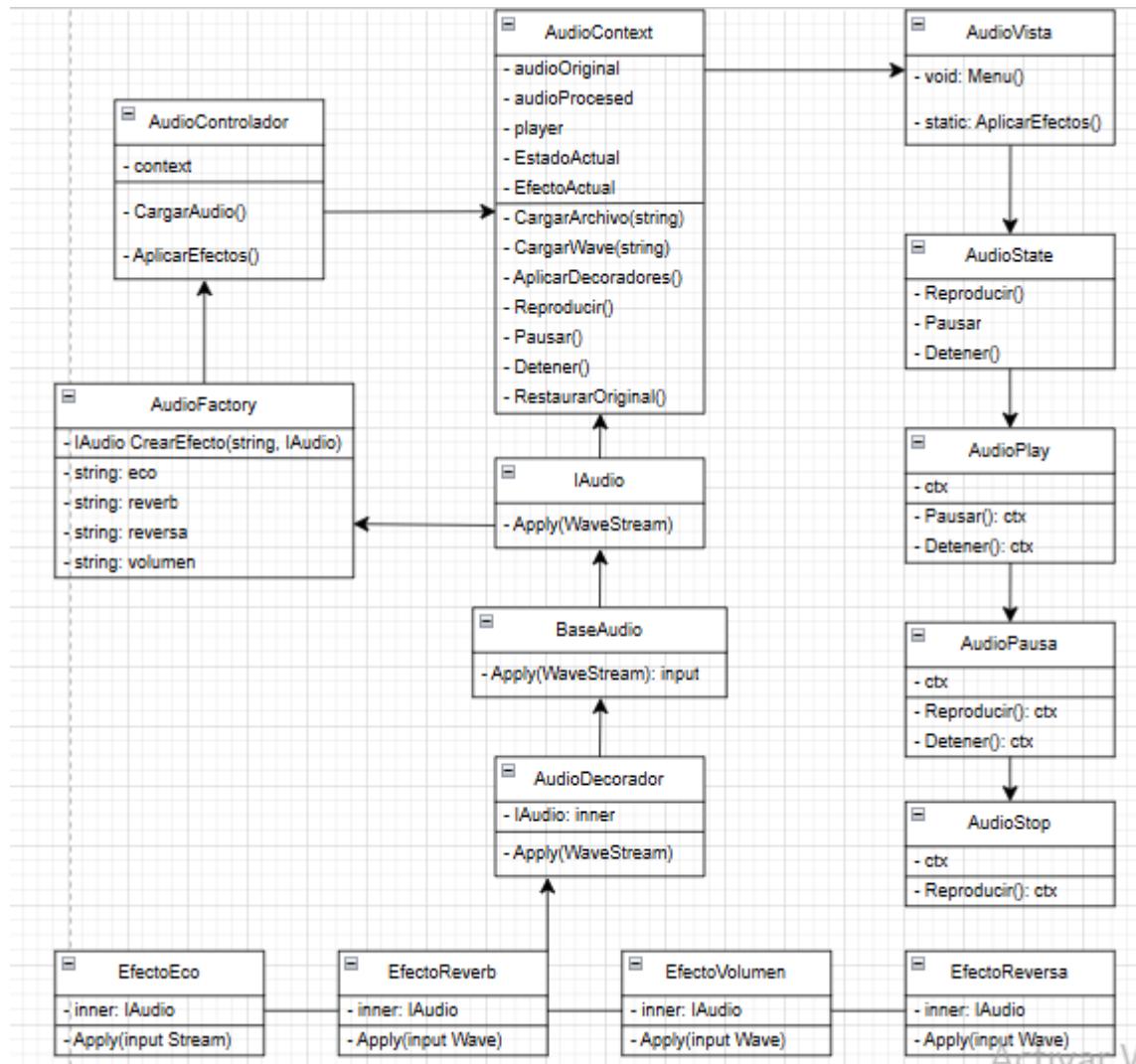
NOMBRE Y NÚMERO DE CONTROL DEL ALUMNO:  
Gonzalez Vazquez Melanie Estefania -21211956-

NOMBRE DEL DOCENTE:  
Maribel Guerrero Luis

LUGAR Y FECHA:  
Tijuana, Baja California, México a 11 de Diciembre del 2025



## Diagrama UML





## Código

### Program.cs

```
1  using Examen_U4_Patrones.Controlador;
2  using System;
3  using System.Collections.Generic;
4  using System.Linq;
5  using System.Text;
6  using System.Threading.Tasks;
7  using Examen_U4_Patrones.Vista;
8
9  namespace Examen_U4_Patrones
10 {
11     public class Program
12     {
13         static void Main(string[] args)
14         {
15             AudioVista.Menu();
16         }
17     }
18 }
```

### Controlador: AudioControlador.cs

```
6  using Examen_U4_Patrones.Modelo;
7  using Examen_U4_Patrones.Modelo.Decorador;
8  using Examen_U4_Patrones.Modelo.Factory;
9  using Examen_U4_Patrones.Modelo.State;
10 using System.IO;
11
12 namespace Examen_U4_Patrones.Controlador
13 {
14     public class AudioControlador
15     {
16         private AudioContext context = new AudioContext();
17
18         public bool CargarAudio(string ruta)
19         {
20             return context.CargaArchivo(ruta);
21         }
22
23         public void Reproducir() => context.Reproducir();
24         public void Pausar() => context.Pausar();
25         public void Detener() => context.Detener();
26         public void RestaurarOriginal() => context.RestablecerOriginal();
27
28         public void AplicarEfectos(string[] efectos)
29         {
30             context.RestablecerOriginal();
31
32             foreach (string ef in efectos)
33             {
34                 string limpia = ef.Trim();
35                 var decorador = AudioFactory.CrearEfecto(limpia, context.EfectoActual);
36                 context.EfectoActual = decorador;
37             }
38
39             context.AplicarDecoradores();
40         }
41     }
42 }
```

### Modelo: AudioContext.cs

```
7  using Examen_U4_Patrones.Modelo.Decorador;
8  using Examen_U4_Patrones.Modelo.State;
9  using NAudio.Wave;
10
11 namespace Examen_U4_Patrones.Modelo
12 {
13     public class AudioContext
14     {
15         public AudioEstado EstadoActual { get; set; }
16         public IAudio EfectoActual { get; set; } = new BaseAudio();
17
18         private WaveOutEvent player;
19         private WaveStream audioOriginal;
20         private WaveStream audioProcesado;
21         private string archivo;
22
23         public AudioContext()
24         {
25             EstadoActual = new AudioStop(this);
26         }
27
28         public WaveStream AudioStream => audioProcesado;
29
30         public bool CargaArchivo(string ruta)
31         {
32             try
33             {
34                 archivo = ruta;
35                 audioOriginal = Cargarwave(ruta);
36                 audioProcesado = audioOriginal;
37
38                 player = new WaveOutEvent();
39                 player.Init(audioProcesado);
40
41                 return true;
42             }
43         }
44     }
45 }
```



Examen\_U4

```
42     }
43     catch
44     {
45         return false;
46     }
47 }
48
49 //referencia
50 public MemoryStream CargarWave(string ruta)
51 {
52     string ext = System.IO.Path.GetExtension(ruta).ToLower();
53
54     if (ext == ".wma")
55         return new WaveFileReader(ruta);
56     else if (ext == ".mp3")
57         return new Mp3FileReader(ruta);
58     else
59         throw new Exception("Formato invalido.");
60 }
61
62 //referencia
63 public void AplicarDecoradores()
64 {
65     audioProcesado = efectoActual.Apply(audioOriginal);
66     player.Stop();
67     player.Init(audioProcesado);
68 }
69
70 //referencia
71 public void Reproducir() => EstadoActual.Reproducir();
72 //referencia
73 public void Pausar() => EstadoActual.Pausar();
74 //referencia
75 public void Detener() => EstadoActual.Detener();
76
77 //referencia
78 public void Play() => player.Play();
79 //referencia
80 public void Pause() => player.Pause();
81 //referencia
82 public void Stop() => player.Stop();
83
84 //referencia
85 public void RestablecerOriginal()
86 {
87     EfectoActual = new Basico();
88     audioProcesado = audioOriginal;
89     player.Init(audioProcesado);
90 }
```

Buscar en Explorador de soluc

Solución "Examen\_U4" (1 de 1)

- Examen\_U4
  - Properties
  - Referencias
  - Controlador
  - Modelo
    - Decorador
    - Factory
    - State
  - C# AudioContext.cs
  - Vista
  - App.config
  - packages.config
  - C# Program.cs

## Modelo: State: AudioEstado.cs

Examen\_U4

```
1  using System;
2  using System.Collections.Generic;
3  using System.Linq;
4  using System.Text;
5  using System.Threading.Tasks;
6
7  namespace Examen_U4_Patrones.Modelo.State
8  {
9      4 referencias
10     public interface AudioEstado
11     {
12         4 referencias
13         void Reproducir();
14         4 referencias
15         void Pausar();
16         4 referencias
17         void Detener();
18     }
19 }
```

Buscar en Explorador de soluc

Solución "Examen\_U4" (1 de 1)

- Examen\_U4
  - Properties
  - Referencias
  - Controlador
  - Modelo
    - Decorador
    - Factory
    - State
      - C# AudioEstado.cs
      - C# AudioPausa.cs
      - C# AudioPlay.cs
      - C# AudioStop.cs
  - C# AudioContext.cs
  - Vista
  - App.config
  - packages.config
  - C# Program.cs

## Modelo: State: AudioPausa.cs

Examen\_U4

```
1  using Examen_U4_Patrones.Controlador;
2  using System;
3  using System.Collections.Generic;
4  using System.Linq;
5  using System.Text;
6  using System.Threading.Tasks;
7
8  namespace Examen_U4_Patrones.Modelo.State
9  {
10     2 referencias
11     public class AudioPausa : AudioEstado
12     {
13         private AudioContext ctx;
14
15         1 referencia
16         public AudioPausa(AudioContext c) => ctx = c;
17
18         2 referencias
19         public void Reproducir()
20         {
21             ctx.Play();
22             ctx.EstadoActual = new AudioPlay(ctx);
23         }
24
25         2 referencias
26         public void Pausar()
27         {
28             ctx.Stop();
29             ctx.EstadoActual = new AudioStop(ctx);
30         }
31     }
32 }
```

Buscar en Explorador de soluc

Solución "Examen\_U4" (1 de 1)

- Examen\_U4
  - Properties
  - Referencias
  - Controlador
  - Modelo
    - Decorador
    - Factory
    - State
      - C# AudioEstado.cs
      - C# AudioPausa.cs
      - C# AudioPlay.cs
      - C# AudioStop.cs
  - C# AudioContext.cs
  - Vista
  - App.config
  - packages.config
  - C# Program.cs



## Modelo: State: AudioPlay.cs

The screenshot shows the Visual Studio IDE with the solution 'Examen\_U4' open. The code editor displays the file 'Examen\_U4\_Patrones.Modelo.State.AudioPlay.cs'. The code implements the State pattern for audio playback:

```
1  using Examen_U4_Patrones.Controlador;
2  using System;
3  using System.Collections.Generic;
4  using System.Linq;
5  using System.Text;
6  using System.Threading.Tasks;
7
8  namespace Examen_U4_Patrones.Modelo.State
9  {
10    public class AudioPlay : AudioEstado
11    {
12      private AudioContext ctx;
13
14      public AudioPlay(AudioContext c) => ctx = c;
15
16      public void Reproducir()
17      {
18        ctx.Pause();
19        ctx.EstadoActual = new AudioPausa(ctx);
20      }
21
22      public void Pausar()
23      {
24        ctx.Stop();
25        ctx.EstadoActual = new AudioStop(ctx);
26      }
27
28    }
29 }
```

The Solution Explorer on the right shows the project structure:

- Examen\_U4
- Properties
- Referencias
- Controlador
- Modelo
  - Decorador
  - Factory
  - State
    - AudioEstado.cs
    - AudioPausa.cs
    - AudioPlay.cs
    - AudioStop.cs
- Vista
- App.config
- packages.config
- Program.cs

## Modelo: State: AudioStop.cs

The screenshot shows the Visual Studio IDE with the solution 'Examen\_U4' open. The code editor displays the file 'Examen\_U4\_Patrones.Modelo.State.AudioStop.cs'. The code implements the State pattern for audio stopping:

```
1  using Examen_U4_Patrones.Controlador;
2  using System;
3  using System.Collections.Generic;
4  using System.Linq;
5  using System.Text;
6  using System.Threading.Tasks;
7
8  namespace Examen_U4_Patrones.Modelo.State
9  {
10    public class AudioStop : AudioEstado
11    {
12      private AudioContext ctx;
13
14      public AudioStop(AudioContext c) => ctx = c;
15
16      public void Reproducir()
17      {
18        ctx.Play();
19        ctx.EstadoActual = new AudioPlay(ctx);
20      }
21
22      public void Pausar() {}
23
24      public void Detener() {}
25
26    }
27 }
```

The Solution Explorer on the right shows the project structure:

- Examen\_U4
- Properties
- Referencias
- Controlador
- Modelo
  - Decorador
  - Factory
  - State
    - AudioEstado.cs
    - AudioPausa.cs
    - AudioPlay.cs
    - AudioStop.cs
- Vista
- App.config
- packages.config
- Program.cs

## Modelo: Factory: AudioFactory.cs

The screenshot shows the Visual Studio IDE with the solution 'Examen\_U4' open. The code editor displays the file 'Examen\_U4\_Patrones.Modelo.Factory.AudioFactory.cs'. The code implements a factory for creating audio effects:

```
1  using System.Text;
2  using System.Threading.Tasks;
3  using Examen_U4_Patrones.Modelo.Decorador;
4  using Examen_U4_Patrones.Modelo.Efecto;
5
6  namespace Examen_U4_Patrones.Modelo.Factory
7  {
8    public class AudioFactory
9    {
10      public static IAudio CrearEfecto(string tipo, IAudio baseEffect)
11      {
12        tipo = tipo.ToLower();
13
14        switch(tipo)
15        {
16          case "eco":
17            return new EfectoEco(baseEffect);
18
19          case "reverb":
20            return new EfectoReverb(baseEffect);
21
22          case "volumen":
23            return new EfectoVolumen(baseEffect);
24
25          case "reversa":
26            return new EfectoReversa(baseEffect);
27
28          default:
29            return baseEffect;
30        }
31      }
32
33    }
34  }
```

The Solution Explorer on the right shows the project structure:

- Examen\_U4
- Properties
- Referencias
- Controlador
- Modelo
  - Decorador
  - Factory
    - AudioFactory.cs
    - State
    - AudioContext.cs
  - Vista
  - App.config
  - packages.config
  - Program.cs



## Modelo: Decorador: IAudio.cs

The screenshot shows a Visual Studio interface with the following details:

- Solution Explorer:** Shows the solution "Examen\_U4" with the following structure:
  - Properties
  - Referencias
  - Controlador
  - Modelo
    - Decorador
      - C# AudioDecorador.cs
      - C# BaseAudio.cs
      - C# EfectoEco.cs
      - C# EfectoReverb.cs
      - C# EfectoReversa.cs
      - C# EfectoVolumen.cs
      - C# IAudio.cs
    - Factory
    - State
    - C# AudioContext.cs
  - Vista
  - App.config
  - packages.config
  - C# Program.cs
- Code Editor:** The file "IAudio.cs" is open, containing the following C# code:

```
1 using System;
2 using System.Collections.Generic;
3 using System.Linq;
4 using System.Text;
5 using System.Threading.Tasks;
6 using NAudio.Wave;
7
8 namespace Examen_U4_Patrones.Modelo.Dekorador
9 {
10     public interface IAudio
11     {
12         WaveStream Apply(WaveStream input);
13     }
14 }
```

## Modelo: Decorador: BaseAudio.cs

The screenshot shows a Visual Studio interface with the following details:

- Solution Explorer:** Shows the solution "Examen\_U4" with the following structure:
  - Properties
  - Referencias
  - Controlador
  - Modelo
    - Decorador
      - C# AudioDecorador.cs
      - C# BaseAudio.cs
      - C# EfectoEco.cs
      - C# EfectoReverb.cs
      - C# EfectoReversa.cs
      - C# EfectoVolumen.cs
      - C# IAudio.cs
    - Factory
    - State
    - C# AudioContext.cs
  - Vista
  - App.config
  - packages.config
  - C# Program.cs
- Code Editor:** The file "BaseAudio.cs" is open, containing the following C# code:

```
1 using System;
2 using System.Collections.Generic;
3 using System.Linq;
4 using System.Text;
5 using System.Threading.Tasks;
6 using NAudio.Wave;
7
8 namespace Examen_U4_Patrones.Modelo.Dekorador
9 {
10     public class BaseAudio : IAudio
11     {
12         public WaveStream Apply(WaveStream input) => input;
13     }
14 }
```

## Modelo: Decorador: AudioDecorador.cs

The screenshot shows a Visual Studio interface with the following details:

- Solution Explorer:** Shows the solution "Examen\_U4" with the following structure:
  - Properties
  - Referencias
  - Controlador
  - Modelo
    - Decorador
      - C# AudioDecorador.cs
      - C# BaseAudio.cs
      - C# EfectoEco.cs
      - C# EfectoReverb.cs
      - C# EfectoReversa.cs
      - C# EfectoVolumen.cs
      - C# IAudio.cs
    - Factory
    - State
    - C# AudioContext.cs
  - Vista
  - App.config
  - packages.config
  - C# Program.cs
- Code Editor:** The file "AudioDecorador.cs" is open, containing the following C# code:

```
1 using System;
2 using System.Collections.Generic;
3 using System.Linq;
4 using System.Text;
5 using System.Threading.Tasks;
6 using NAudio.Wave;
7
8 namespace Examen_U4_Patrones.Modelo.Dekorador
9 {
10     public abstract class AudioDecorador : IAudio
11     {
12         protected IAudio inner;
13
14         protected AudioDecorador(IAudio innerAudio)
15         {
16             this.inner = innerAudio;
17         }
18
19         public abstract WaveStream Apply(WaveStream input);
20     }
21 }
22 }
```



## Modelo: Decorador: EfectoEco.cs

The screenshot shows the Visual Studio IDE with the code editor open to the file `EfectoEco.cs`. The code implements the `WaveStream` interface by applying a delay effect to an input stream. It uses various `NAudio.Wave` classes like `OffsetSampleProvider`, `VolumeSampleProvider`, and `MixerSampleProvider`. The Solution Explorer on the right shows the project structure for `Examen_U4`, including files like `BaseAudio.cs`, `IAudio.cs`, and other effect classes.

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using NAudio.Wave;
using NAudio.Wave.SampleProviders;

namespace Examen_U4_Patrones.Modelo.Decorador
{
    2 referencias
    public class EfectoEco : AudioDecorador
    {
        1 referencia
        public EfectoEco(IAudio inner) : base(inner) {}

        6 referencias
        public override WaveStream Apply(WaveStream input)
        {
            var original = input.ToSampleProvider();
            var delayedSource = Input.ToSampleProvider();
            var delay = new OffsetSampleProvider(delayedSource)
            {
                DelayBy = TimeSpan.FromMilliseconds(300)
            };
            var delayVol = new VolumeSampleProvider(delay)
            {
                Volume = 0.7f
            };

            var mixer = new MixingSampleProvider(Microsoft.Xna.Framework.CreateIeeeFloatWaveFormat(
                input.WaveFormat.SampleRate,
                input.WaveFormat.Channels));
            mixer.AddMixerInput(original);
            mixer.AddMixerInput(new VolumeSampleProvider(delay)
            {
                Volume = 0.4f
            });

            return mixer.ToWaveProvider().ToWaveStream();
        }

        6 referencias
        public static WaveStream ToWaveStream(this IWaveProvider wave)
        {
            return new RawSourceWaveStream((Stream)wave, wave.WaveFormat);
        }
    }
}
```

## Modelo: Decorador: EfectoReverb.cs

The screenshot shows the Visual Studio IDE with the code editor open to the file `EfectoReverb.cs`. This class applies a reverb effect by reversing the input data. The Solution Explorer on the right shows the project structure for `Examen_U4`, including files like `BaseAudio.cs`, `IAudio.cs`, and other effect classes.

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using NAudio.Wave;
using NAudio.Wave.SampleProviders;

namespace Examen_U4_Patrones.Modelo.Decorador
{
    2 referencias
    public class EfectoReverb : AudioDecorador
    {
        1 referencia
        public EfectoReverb(IAudio inner) : base(inner) {}

        6 referencias
        public override WaveStream Apply(WaveStream input)
        {
            var processed = inner.Apply(input);

            processed.Position = 0;
            byte[] data = new byte[processed.Length];
            processed.Read(data, 0, data.Length);
            Array.Reverse(data);

            return new RawSourceWaveStream(data, 0, data.Length, processed.WaveFormat);
        }
    }
}
```

## Modelo: Decorador: EfectoReversa.cs

The screenshot shows the Visual Studio IDE with the code editor open to the file `EfectoReversa.cs`. This class applies a reverse effect by reading the input data from the end and writing it to the beginning. The Solution Explorer on the right shows the project structure for `Examen_U4`, including files like `BaseAudio.cs`, `IAudio.cs`, and other effect classes.

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using NAudio.Wave;
using NAudio.Wave.SampleProviders;

namespace Examen_U4_Patrones.Modelo.Decorador
{
    2 referencias
    public class EfectoReversa : AudioDecorador
    {
        1 referencia
        public EfectoReversa(IAudio inner) : base(inner) {}

        6 referencias
        public override WaveStream Apply(WaveStream input)
        {
            var processed = inner.Apply(input);
            var sample = processed.ToSampleProvider();

            var delay1 = new OffsetSampleProvider(sample) { DelayBy = TimeSpan.FromMilliseconds(0) };
            var delay2 = new OffsetSampleProvider(sample) { DelayBy = TimeSpan.FromMilliseconds(80) };
            var delay3 = new OffsetSampleProvider(sample) { DelayBy = TimeSpan.FromMilliseconds(120) };

            var d1 = new VolumeSampleProvider(delay1) { Volume = 0.4f };
            var d2 = new VolumeSampleProvider(delay2) { Volume = 0.25f };
            var d3 = new VolumeSampleProvider(delay3) { Volume = 0.15f };

            var mixer = new MixingSampleProvider(sample.WaveFormat)
            {
                ReadFully = true
            };
            mixer.AddMixerInput(sample);
            mixer.AddMixerInput(d1);
            mixer.AddMixerInput(d2);
            mixer.AddMixerInput(d3);
        }
    }
}
```



```
Examen_U4
Examen_U4_Patrones.Modelo.Decorador
WaveFormat

37    }
38    }
39    }
40    }
41    }
42    }
43    }
44    }
45    }
46    }
47    }
48    }
49    }
50    }
51    }
52    }
53    }
54    }
55    }
56    }
57    }
58    }
59    }
60    }
61    }
62    }
63    }
64    }
65    }
66    }
67    }
68    }
69    }
70    }
71    }

1 referencia
public class SampleProviderToWaveStream : WaveStream
{
    private readonly ISampleProvider source;
    private readonly WaveFormat waveFormat;
    private long position;

    1 referencia
    public SampleProviderToWaveStream(ISampleProvider source)
    {
        this.source = source;
        this.waveFormat = source.WaveFormat;
    }

    0 referencias
    public override WaveFormat WaveFormat => waveFormat;

    0 referencias
    public override long Length => long.MaxValue;

    0 referencias
    public override long Position
    {
        get => position;
        set => position = value;
    }

    0 referencias
    public override int Read(byte[] buffer, int offset, int count)
    {
        int samplesRequired = count / 4;
        float[] sampleBuffer = new float[samplesRequired];

        int samplesRead = source.Read(sampleBuffer, 0, samplesRequired);

        Buffer.BlockCopy(sampleBuffer, 0, buffer, offset, samplesRead * 4);
    }
}
```

## Modelo: Decorador: EfectoVolumen.cs

```
Examen_U4
Examen_U4_Patrones.Modelo.Decorador
Apply(WaveStream input)

1   using System;
2   using System.Collections.Generic;
3   using System.Linq;
4   using System.Text;
5   using System.Threading.Tasks;
6   using NAudio.Wave;
7   using NAudio.Wave.SampleProviders;
8
9   namespace Examen_U4_Patrones.Modelo.Decorador
10  {
11      2 referencias
12      public class EfectoVolumen : AudioDecorador
13      {
14          1 referencia
15          public EfectoVolumen(IAudio inner) : base(inner) { }
16
17          6 referencias
18          public override WaveStream Apply(WaveStream input)
19          {
20              var processed = inner.Apply(input);
21              var vol = new WaveChannel32(processed);
22              vol.Volume = 0.5f;
23              return vol;
24          }
25      }
26  }
```

## Modelo: Vista: AudioVista.cs

```
Examen_U4
Examen_U4_Patrones.Controlador;
Examen_U4_Patrones.Modelo;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using Examen_U4_Patrones.Vista
namespace Examen_U4_Patrones.Vista
{
    1 referencia
    public static class AudioVista
    {
        1 referencia
        public static void Menu()
        {
            AudioControlador controller = new AudioControlador();
            //AudioContext context = new AudioContext();

            Console.WriteLine("==> GESTOR DE AUDIO ==>");
            Console.WriteLine("Ingresá ruta del archivo (.wav o .mp3): ");
            string ruta = Console.ReadLine();

            if (!controller.CargarAudio(ruta))
            {
                Console.WriteLine("No se pudo cargar el archivo.");
                return;
            }

            while (true)
            {
                Console.WriteLine("\n==> MENÚ ==>");
                Console.WriteLine("(1) Reproducir");
                Console.WriteLine("(2) Pausar");
                Console.WriteLine("(3) Detener");
                Console.WriteLine("(4) Aplicar efectos (mezclador)");
                Console.WriteLine("(5) Reproducir audio original");
                Console.WriteLine("(6) Salir");
                Console.WriteLine("Opción: ");
                string op = Console.ReadLine();
            }
        }
    }
}
```



The screenshot shows the Microsoft Visual Studio interface. The code editor on the left contains C# code for a class named `Vista`. The Solution Explorer on the right shows a solution named "Examen\_U4" containing projects for `Properties`, `Referencias`, `Controlador`, `Modelo`, and `Vista`. The `Vista` project is expanded, showing files like `AudioVista.cs`, `App.config`, `packages.config`, and `Program.cs`.

```
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
```

```
switch (op)
{
    case "1": controller.Reproducir(); break;
    case "2": controller.Pausar(); break;
    case "3": controller.Detener(); break;
    case "4": AplicarEfectos(controller); break;
    case "5": controller.RestaurarOriginal(); break;
    case "6": return;
    default: Console.WriteLine("Opción inválida"); break;
}

1 referencia
static void AplicarEfectos(AudioControlador controller)
{
    Console.WriteLine("\nSelecciona efectos separados por coma:");
    Console.WriteLine("eco, reverb, volumen, reversa");
    Console.Write("Ejemplo: eco, volumen: ");

    string entrada = Console.ReadLine();
    string[] lista = entrada.ToLower().Split(',');
    controller.AplicarEfectos(lista);
    Console.WriteLine("Efectos aplicados.");
}
```

## Ejecución

```
C:\Users\dell\source\repos\Examen_U4\Examen_U4\bin\Debug\Examen_U4.exe
== GESTOR DE AUDIO ==
Ingresa ruta del archivo (.wav o .mp3): C:\Users\dell\Music\luna.mp3
```

```
C:\Users\dell\source\repos\Examen_U4\Examen_U4\bin\Debug\Examen_U4.exe
== GESTOR DE AUDIO ==
Ingresa ruta del archivo (.wav o .mp3): C:\Users\dell\Music\luna.mp3

== MENÚ ==
1) Reproducir
2) Pausar
3) Detener
4) Aplicar efectos (mezclador)
5) Restaurar audio original
6) Salir
Opción: 1

== MENÚ ==
1) Reproducir
2) Pausar
3) Detener
4) Aplicar efectos (mezclador)
5) Restaurar audio original
6) Salir
Opción: -
```



```
== MENÚ ==
1) Reproducir
2) Pausar
3) Detener
4) Aplicar efectos (mezclador)
5) Restaurar audio original
6) Salir
Opción: 2

== MENÚ ==
1) Reproducir
2) Pausar
3) Detener
4) Aplicar efectos (mezclador)
5) Restaurar audio original
6) Salir
Opción: 3

== MENÚ ==
1) Reproducir
2) Pausar
3) Detener
4) Aplicar efectos (mezclador)
5) Restaurar audio original
6) Salir
Opción: _
```

```
1) Reproducir
2) Pausar
3) Detener
4) Aplicar efectos (mezclador)
5) Restaurar audio original
6) Salir
Opción: 4
```

Selecciona efectos separados por coma:  
eco, reverb, volumen, reversa  
Ejemplo: eco, volumen: reversa  
Efectos aplicados.

```
== MENÚ ==
1) Reproducir
2) Pausar
3) Detener
4) Aplicar efectos (mezclador)
5) Restaurar audio original
6) Salir
Opción: 1
```

```
== MENÚ ==
1) Reproducir
2) Pausar
3) Detener
4) Aplicar efectos (mezclador)
5) Restaurar audio original
6) Salir
Opción: _
```

```
== MENÚ ==
1) Reproducir
2) Pausar
3) Detener
4) Aplicar efectos (mezclador)
5) Restaurar audio original
6) Salir
Opción: 5

== MENÚ ==
1) Reproducir
2) Pausar
3) Detener
4) Aplicar efectos (mezclador)
5) Restaurar audio original
6) Salir
Opción: 1

== MENÚ ==
1) Reproducir
2) Pausar
3) Detener
4) Aplicar efectos (mezclador)
5) Restaurar audio original
6) Salir
Opción: _
```