**Задание**

Создать видеоплеер, который позволяет пользователю выбирать фильм и указывать язык звуковой дорожки и субтитров. Язык аудио и субтитров должен совпадать. Система должна воспроизводить фильм с требуемыми характеристиками, обеспечивая согласованность между аудио и субтитрами.

**Реализация без паттерна:**

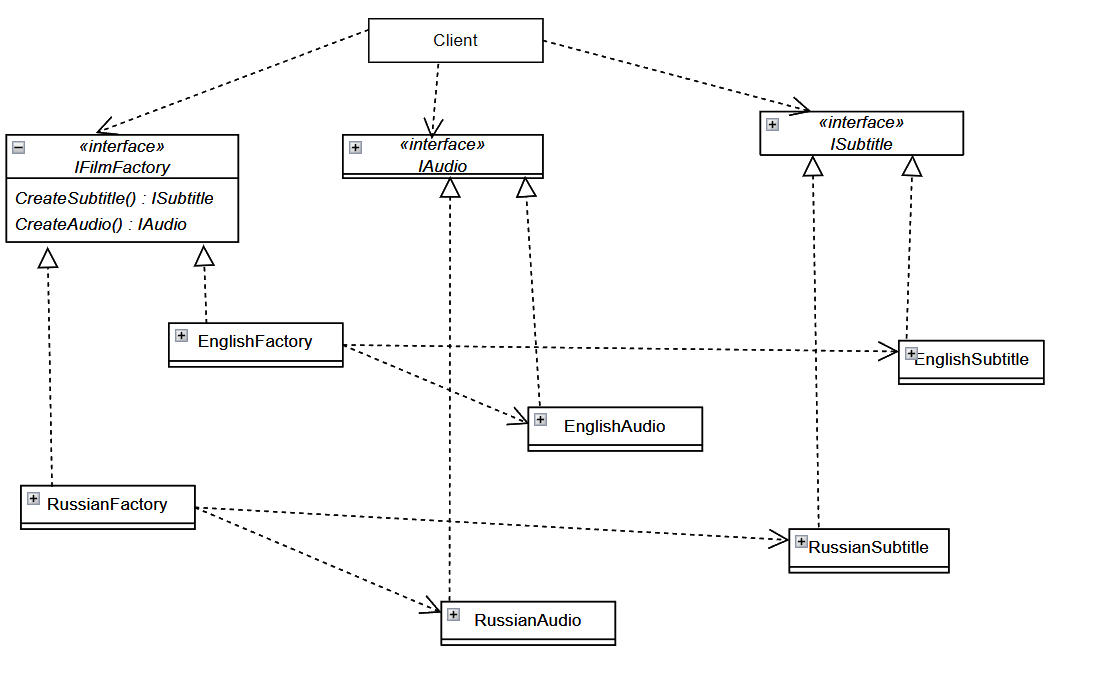
В реализации без паттерна использовался класс AudSub. Этот класс напрямую применял аудио и субтитры к медиаплееру в зависимости от выбранного языка. Код выглядит следующим образом:

|  |
| --- |
| ﻿public class AudSub  {  public static void Audio(string language, LibVLCSharp.Shared.MediaPlayer mediaPlayer, string moviePath)  {  string audioPath = System.IO.Path.Combine(System.IO.Path.GetDirectoryName(moviePath), $"{language}.mp3");  mediaPlayer.Media.AddOption($":input-slave={audioPath}");    }  public static void Subtitle(string language, LibVLCSharp.Shared.MediaPlayer mediaPlayer, string moviePath)  {  string subtitlePath = System.IO.Path.Combine(System.IO.Path.GetDirectoryName(moviePath), $"{language}.srt");  mediaPlayer.Media.AddOption($":sub-file={subtitlePath}");  }  }  private void button3\_Click(object sender, EventArgs e)  {  if (string.IsNullOrEmpty(filePath)) return;  if (mPlayer.IsPlaying) mPlayer.Stop();  var media = new Media(mlibVLC, filePath);  mPlayer.Media = media;  string slang = comboBox1.SelectedItem.ToString();  AudSub.Audio(slang, mPlayer, filePath);  AudSub.Subtitle(slang, mPlayer, filePath);  mPlayer.Play();  } |

Проблема реализации без паттерна заключается в согласованности между аудио и субтитрами. В текущей реализации нет гарантии, что выбранный язык аудио будет соответствовать языку субтитров. Также реализация имеет сложности с расширением.

Абстрактная фабрика решает эти проблемы. Он позволяет легко добавлять новые языки, создавая новые фабрики и продукты. Это достигается за счет создания объектов в отдельных фабриках, каждая из которых отвечает за создание согласованных продуктов для конкретного языка. Таким образом, паттерн гарантирует, что аудио и субтитры будут соответствовать выбранному языку, Это делает систему более удобной для расширения.

**Диаграмма классов реализованного паттерна:**

****

Интерфейс IFilmFactory определяет методы для создания объектов IAudio и ISubtitle, которые представляют аудио и субтитры соответственно. Конкретные фабрики EnglishFilmFactory и RussianFilmFactory реализуют этот интерфейс и создают конкретные продукты: EnglishAudio, RussianAudio, EnglishSubtitle и RussianSubtitle. Каждый из этих продуктов реализует свои интерфейсы и содержит логику для применения аудио или субтитров к медиаплееру.

**Реализация с паттерном:**

|  |
| --- |
| ﻿using System;  public interface IFilmFactory  {  ISubtitle CreateSubtitle();  IAudio CreateAudio();  }  public interface ISubtitle  {  void ApplySubtitle(LibVLCSharp.Shared.MediaPlayer mediaPlayer, string moviePath);  }  public interface IAudio  {  void ApplyAudio(LibVLCSharp.Shared.MediaPlayer mediaPlayer, string moviePath);  }  public class EnglishSubtitle : ISubtitle  {  public void ApplySubtitle(LibVLCSharp.Shared.MediaPlayer mediaPlayer, string moviePath)  {  string subtitlePath = System.IO.Path.Combine(System.IO.Path.GetDirectoryName(moviePath), "English.srt");  mediaPlayer.Media.AddOption($":sub-file={subtitlePath}");  }  }  public class RussianSubtitle : ISubtitle  {  public void ApplySubtitle(LibVLCSharp.Shared.MediaPlayer mediaPlayer, string moviePath)  {  string subtitlePath = System.IO.Path.Combine(System.IO.Path.GetDirectoryName(moviePath), "Russian.srt");  mediaPlayer.Media.AddOption($":sub-file={subtitlePath}");  }  }  public class EnglishAudio : IAudio  {  public void ApplyAudio(LibVLCSharp.Shared.MediaPlayer mediaPlayer, string moviePath)  {  string audioPath = System.IO.Path.Combine(System.IO.Path.GetDirectoryName(moviePath), "English.mp3");  mediaPlayer.Media.AddOption($":input-slave={audioPath}");  }  }  public class RussianAudio : IAudio  {  public void ApplyAudio(LibVLCSharp.Shared.MediaPlayer mediaPlayer, string moviePath)  {  string audioPath = System.IO.Path.Combine(System.IO.Path.GetDirectoryName(moviePath), "Russian.mp3");  mediaPlayer.Media.AddOption($":input-slave={audioPath}");  }  }  public class EnglishFilmFactory : IFilmFactory  {  public ISubtitle CreateSubtitle()  {  return new EnglishSubtitle();  }  public IAudio CreateAudio()  {  return new EnglishAudio();  }  public override string ToString()  {  return "English";  }  }  public class RussianFilmFactory : IFilmFactory  {  public ISubtitle CreateSubtitle()  {  return new RussianSubtitle();  }  public IAudio CreateAudio()  {  return new RussianAudio();  }  public override string ToString()  {  return "Russian";  }  }  private void button2\_Click(object sender, EventArgs e)  {  if (string.IsNullOrEmpty(filePath)) return;  if (mPlayer.IsPlaying) mPlayer.Stop();  var media = new Media(mlibVLC, filePath);  mPlayer.Media = media;  filmFactory = (IFilmFactory)comboBox1.SelectedItem;  var audio = filmFactory.CreateAudio();  var subtitle = filmFactory.CreateSubtitle();  audio.ApplyAudio(mPlayer, filePath);  subtitle.ApplySubtitle(mPlayer, filePath);  mPlayer.Play();  } |