

Refactor

https://www.youtube.com/watch?v=s_4ZrtQs8Do

```
"""
Basic video exporting example
"""

import pathlib
from abc import ABC, abstractmethod

class VideoExporter(ABC):
    """Basic representation of video exporting codec."""

    @abstractmethod
    def prepare_export(self, video_data):
        """Prepares video data for exporting."""

    @abstractmethod
    def do_export(self, folder: pathlib.Path):
        """Exports the video data to a folder."""

class LosslessVideoExporter(VideoExporter):
    """Lossless video exporting codec."""

    def prepare_export(self, video_data):
        print("Preparing video data for lossless export.")

    def do_export(self, folder: pathlib.Path):
        print(f"Exporting video data in lossless format to {folder}.")

class H264BPVideoExporter(VideoExporter):
    """H.264 video exporting codec with Baseline profile."""
```

```
def prepare_export(self, video_data):
    print("Preparing video data for H.264 (Baseline) export.")

def do_export(self, folder: pathlib.Path):
    print(f"Exporting video data in H.264 (Baseline) format to {folder}.")
```

```
class H264Hi422PVideoExporter(VideoExporter):
    """H.264 video exporting codec with Hi422P profile (10-bit, 4:2:2 chroma sampling)."""

    def prepare_export(self, video_data):
        print("Preparing video data for H.264 (Hi422P) export.")

    def do_export(self, folder: pathlib.Path):
        print(f"Exporting video data in H.264 (Hi422P) format to {folder}.")
```

```
class AudioExporter(ABC):
    """Basic representation of audio exporting codec."""

    @abstractmethod
    def prepare_export(self, audio_data):
        """Prepares audio data for exporting."""

    @abstractmethod
    def do_export(self, folder: pathlib.Path):
        """Exports the audio data to a folder."""
```

```
class AACAudioExporter(AudioExporter):
    """AAC audio exporting codec."""

    def prepare_export(self, audio_data):
        print("Preparing audio data for AAC export.")

    def do_export(self, folder: pathlib.Path):
        print(f"Exporting audio data in AAC format to {folder}.")
```

```
class WAVAudioExporter(AudioExporter):
```

```
"""WAV (lossless) audio exporting codec."""
```

```
def prepare_export(self, audio_data):  
    print("Preparing audio data for WAV export.")
```

```
def do_export(self, folder: pathlib.Path):  
    print(f"Exporting audio data in WAV format to {folder}.")
```

```
def main() -> None:
```

```
    """Main function."""
```

```
    # read the desired export quality
```

```
    export_quality: str
```

```
    while True:
```

```
        export_quality = input("Enter desired output quality (low, high, master): ")
```

```
        if export_quality in {"low", "high", "master"}:
```

```
            break
```

```
        print(f"Unknown output quality option: {export_quality}.")
```

```
    # create the video and audio exporters
```

```
    video_exporter: VideoExporter
```

```
    audio_exporter: AudioExporter
```

```
    if export_quality == "low":
```

```
        video_exporter = H264BPVideoExporter()
```

```
        audio_exporter = AACAudioExporter()
```

```
    elif export_quality == "high":
```

```
        video_exporter = H264Hi422PVideoExporter()
```

```
        audio_exporter = AACAudioExporter()
```

```
    else:
```

```
        # default: master quality
```

```
        video_exporter = LosslessVideoExporter()
```

```
        audio_exporter = WAVAudioExporter()
```

```
    # prepare the export
```

```
    video_exporter.prepare_export("placeholder_for_video_data")
```

```
    audio_exporter.prepare_export("placeholder_for_audio_data")
```

```
    # do the export
```

```
    folder = pathlib.Path("/usr/tmp/video")
```

```
video_exporter.do_export(folder)
audio_exporter.do_export(folder)
```

```
if __name__ == "__main__":
    main()
```

```
"""
Basic video exporting example
"""

import pathlib
from abc import ABC, abstractmethod

class VideoExporter(ABC):
    """Basic representation of video exporting codec."""

    @abstractmethod
    def prepare_export(self, video_data):
        """Prepares video data for exporting."""

    @abstractmethod
    def do_export(self, folder: pathlib.Path):
        """Exports the video data to a folder."""

class LosslessVideoExporter(VideoExporter):
    """Lossless video exporting codec."""

    def prepare_export(self, video_data):
        print("Preparing video data for lossless export.")

    def do_export(self, folder: pathlib.Path):
        print(f"Exporting video data in lossless format to {folder}.")

class H264BPVideoExporter(VideoExporter):
```

```
"""H.264 video exporting codec with Baseline profile."""
```

```
def prepare_export(self, video_data):  
    print("Preparing video data for H.264 (Baseline) export.")
```

```
def do_export(self, folder: pathlib.Path):  
    print(f"Exporting video data in H.264 (Baseline) format to {folder}.")
```

```
class H264Hi422PVideoExporter(VideoExporter):
```

```
    """H.264 video exporting codec with Hi422P profile (10-bit, 4:2:2 chroma sampling)."""
```

```
def prepare_export(self, video_data):  
    print("Preparing video data for H.264 (Hi422P) export.")
```

```
def do_export(self, folder: pathlib.Path):  
    print(f"Exporting video data in H.264 (Hi422P) format to {folder}.")
```

```
class AudioExporter(ABC):
```

```
    """Basic representation of audio exporting codec."""
```

```
@abstractmethod
```

```
def prepare_export(self, audio_data):  
    """Prepares audio data for exporting."""
```

```
@abstractmethod
```

```
def do_export(self, folder: pathlib.Path):  
    """Exports the audio data to a folder."""
```

```
class AACAudioExporter(AudioExporter):
```

```
    """AAC audio exporting codec."""
```

```
def prepare_export(self, audio_data):  
    print("Preparing audio data for AAC export.")
```

```
def do_export(self, folder: pathlib.Path):  
    print(f"Exporting audio data in AAC format to {folder}.")
```

```

class WAVAudioExporter(AudioExporter):
    """WAV (lossless) audio exporting codec."""

    def prepare_export(self, audio_data):
        print("Preparing audio data for WAV export.")

    def do_export(self, folder: pathlib.Path):
        print(f"Exporting audio data in WAV format to {folder}.")


class ExporterFactory(ABC):
    """
    Factory that represents a combination of video and audio codecs.
    The factory doesn't maintain any of the instances it creates.
    """

    @abstractmethod
    def get_video_exporter(self) -> VideoExporter:
        """Returns a new video exporter belonging to this factory."""

    @abstractmethod
    def get_audio_exporter(self) -> AudioExporter:
        """Returns a new audio exporter belonging to this factory."""


class FastExporter(ExporterFactory):
    """Factory aimed at providing a high speed, lower quality export."""

    def get_video_exporter(self) -> VideoExporter:
        return H264BPVideoExporter()

    def get_audio_exporter(self) -> AudioExporter:
        return AACAudioExporter()


class HighQualityExporter(ExporterFactory):
    """Factory aimed at providing a slower speed, high quality export."""

    def get_video_exporter(self) -> VideoExporter:

```

```

        return H264Hi422PVideoExporter()

def get_audio_exporter(self) -> AudioExporter:
    return AACAudioExporter()

class MasterQualityExporter(ExporterFactory):
    """Factory aimed at providing a low speed, master quality export."""

    def get_video_exporter(self) -> VideoExporter:
        return LosslessVideoExporter()

    def get_audio_exporter(self) -> AudioExporter:
        return WAVAudioExporter()

def read_factory() -> ExporterFactory:
    """Constructs an exporter factory based on the user's preference."""

    factories = {
        "low": FastExporter(),
        "high": HighQualityExporter(),
        "master": MasterQualityExporter(),
    }

    while True:
        export_quality = input("Enter desired output quality (low, high, master): ")
        if export_quality in factories:
            return factories[export_quality]
        print(f"Unknown output quality option: {export_quality}.")

def main(fac: ExporterFactory) -> None:
    """Main function."""

    # retrieve the exporters
    video_exporter = fac.get_video_exporter()
    audio_exporter = fac.get_audio_exporter()

    # prepare the export
    video_exporter.prepare_export("placeholder_for_video_data")

```

```
audio_exporter.prepare_export("placeholder_for_audio_data")

# do the export
folder = pathlib.Path("/usr/tmp/video")
video_exporter.do_export(folder)
audio_exporter.do_export(folder)

if __name__ == "__main__":
    # create the factory
    factory = read_factory()

    # perform the exporting job
    main(factory)
```

Revision #4

Created 3 March 2024 06:15:56 by victor

Updated 3 March 2024 06:21:01 by victor