ORIHALCON Technologies

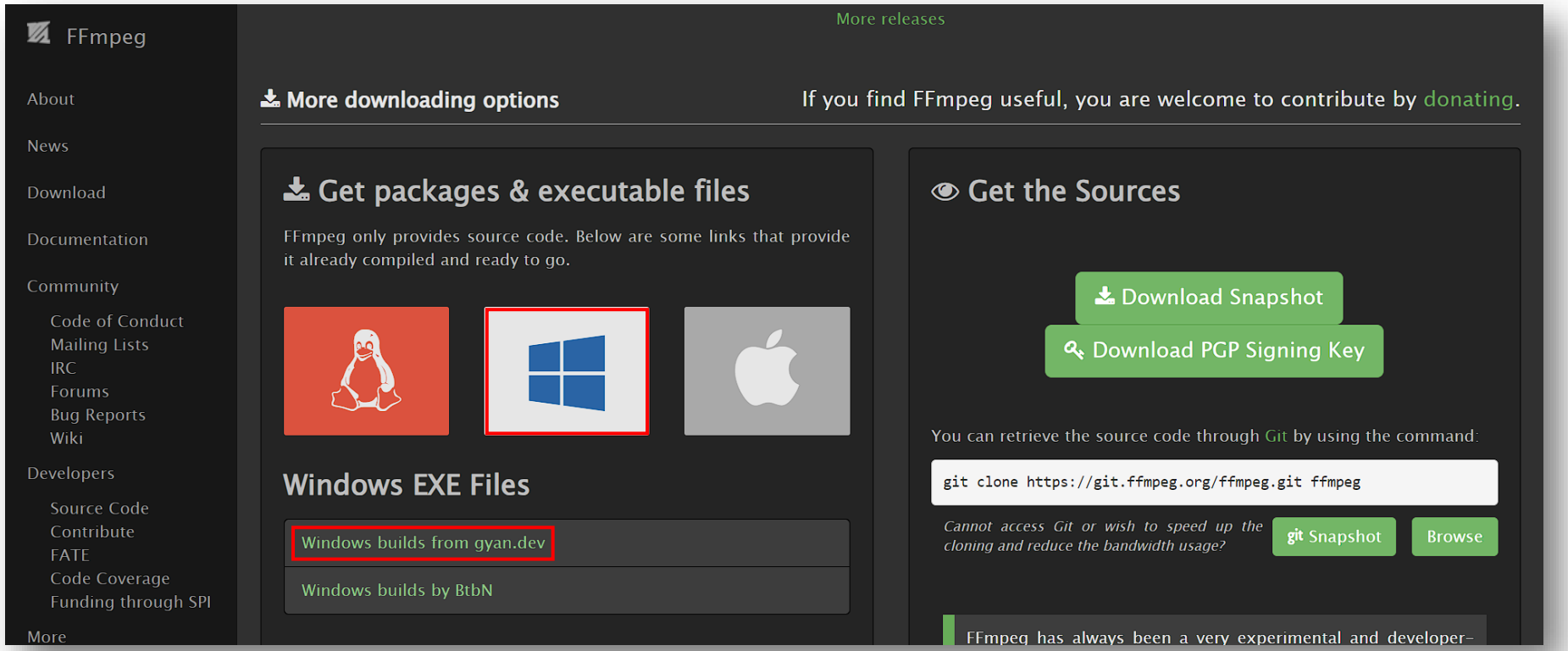
FFmpeg活用法

株式会社オリハルコンテクノロジーズ
上田直哉

1.ダウンロードと展開

FFmpeg公式　https://ffmpeg.org/　にアクセス

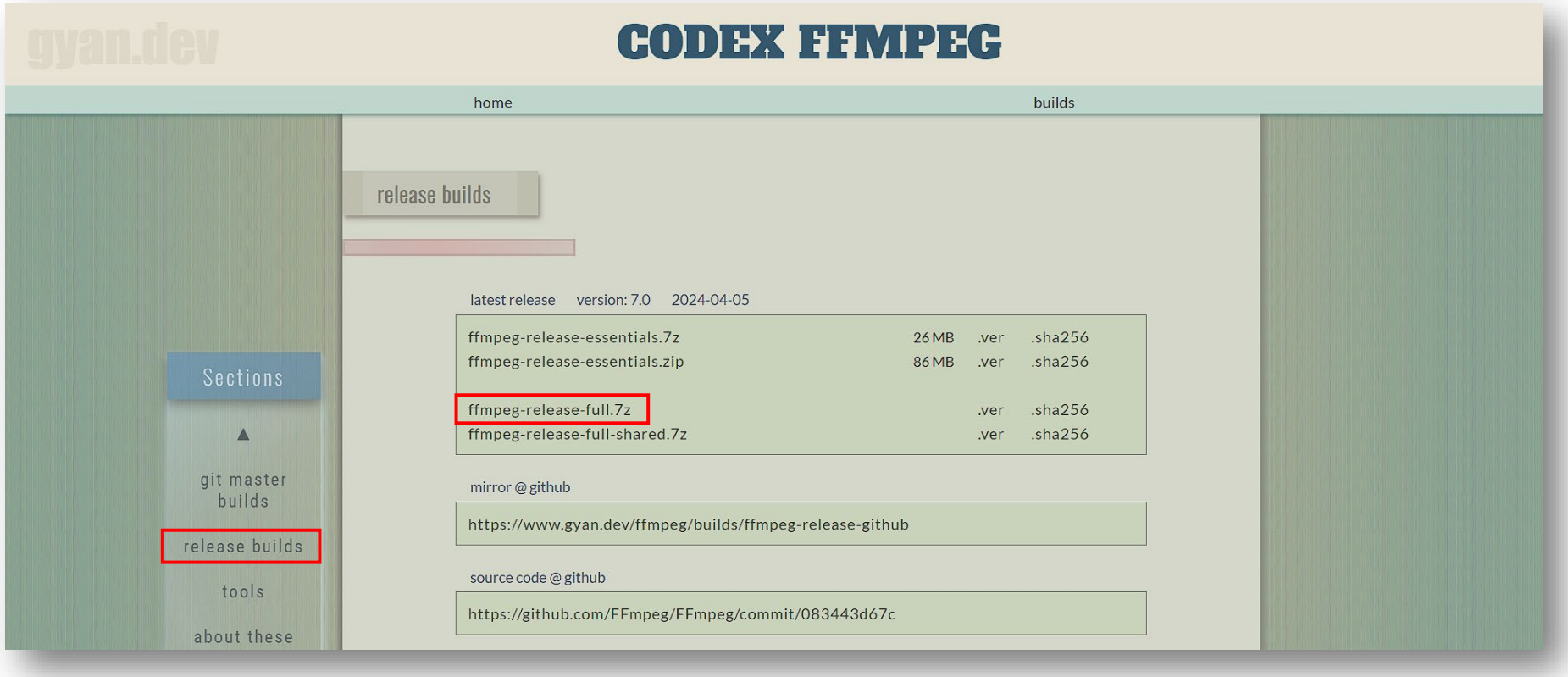
Windowsアイコンをクリックし、 gyan.dev へアクセス



1.ダウンロードと展開

ORIHALCON Technologies

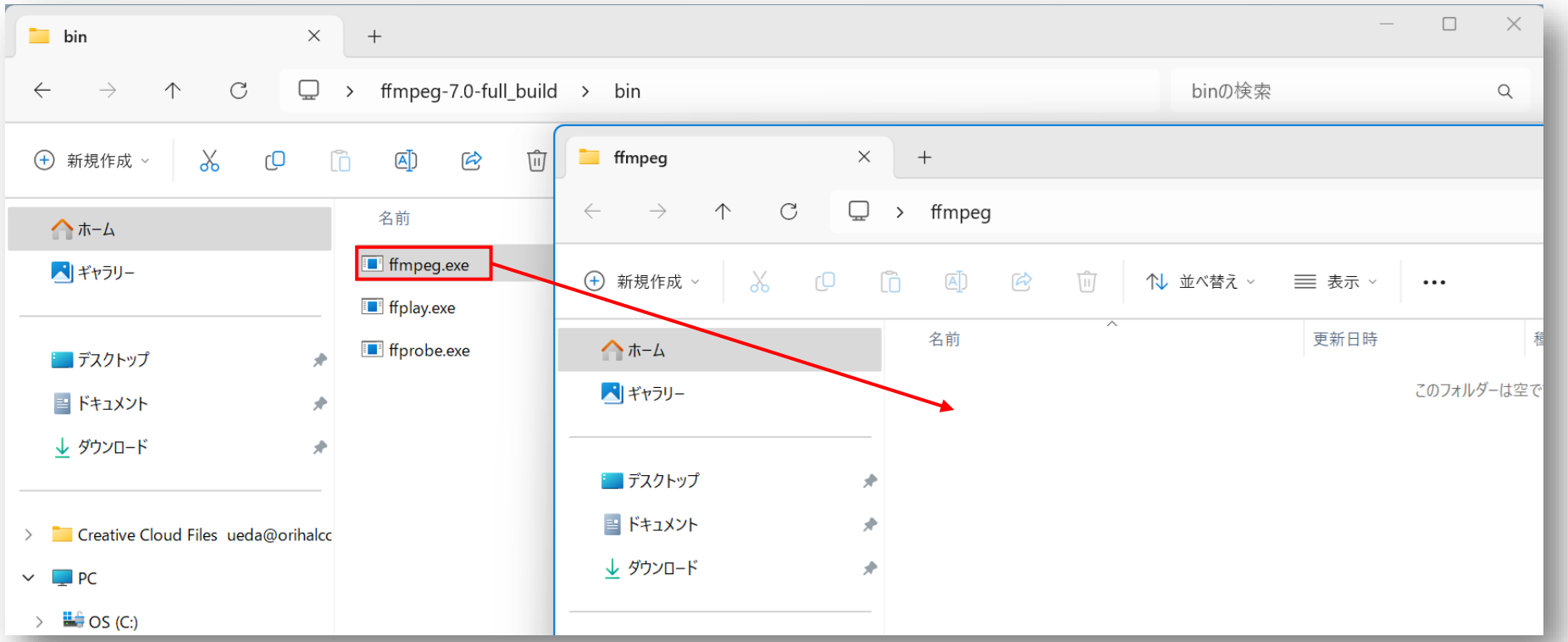
release builds -> latest release のfullをダウンロード



1.ダウンロードと展開

解凍してbinフォルダ内の実行ファイルを取り出す

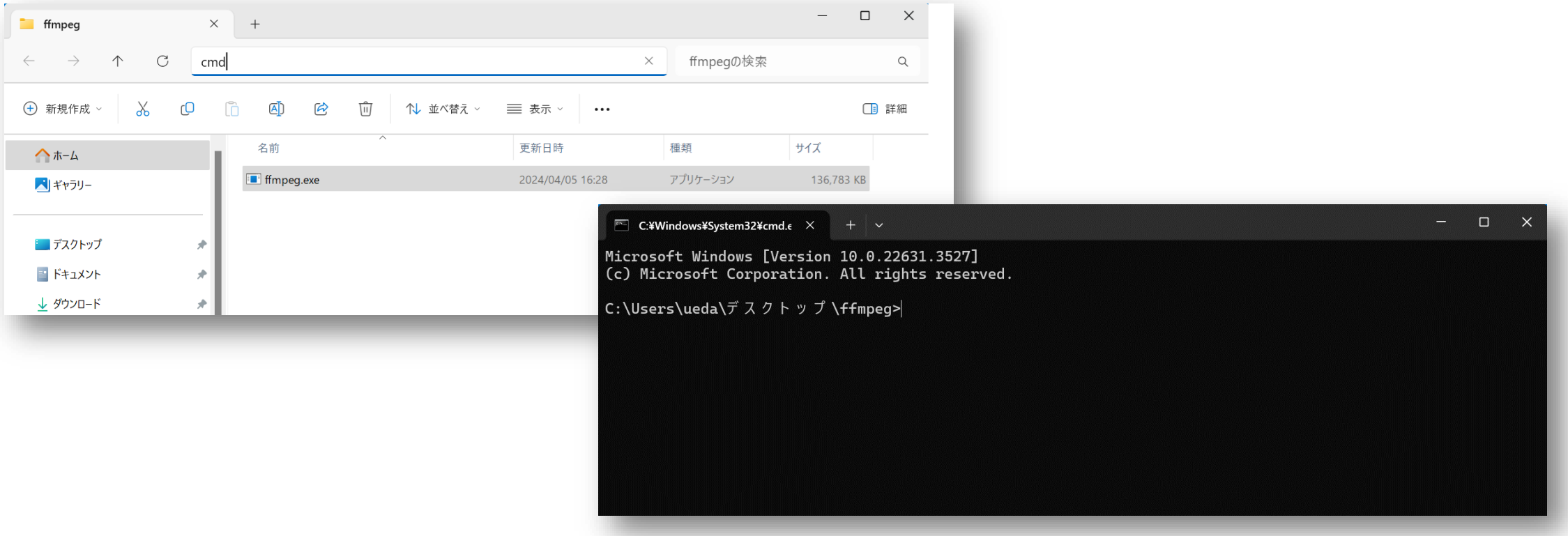
デスクトップにffmpegフォルダを新規作成し、中に実行ファイルを置く



2.ターミナルの起動

FFmpegはGUI（グラフィカルユーザーインターフェース、操作画面）を持たないため、ターミナル（旧コマンドプロンプト）から実行する

エクスプローラのアドレス欄に直接「cmd」と入力するだけで、現在のフォルダをルートにターミナルを起動できる



3.基本的な文法

ffmpeg ＋ -入力オプション ＋ -i 入力ファイル名 ＋ -出力オプション ＋ -フィルタ ＋ 出力ファイル名

-i　　入力ファイルを指定する

出力ファイル名は必ず一番最後に記述する

```
ffmpeg -入力オプション -i INPUT.mp4 -出力オプション –フィルタ OUTPUT.mp4
```

「入力オプションを指定してINPUT.mp4を入力し、出力オプションとフィルタを適用してOUTPUT.mp4を出力」

という意味になる

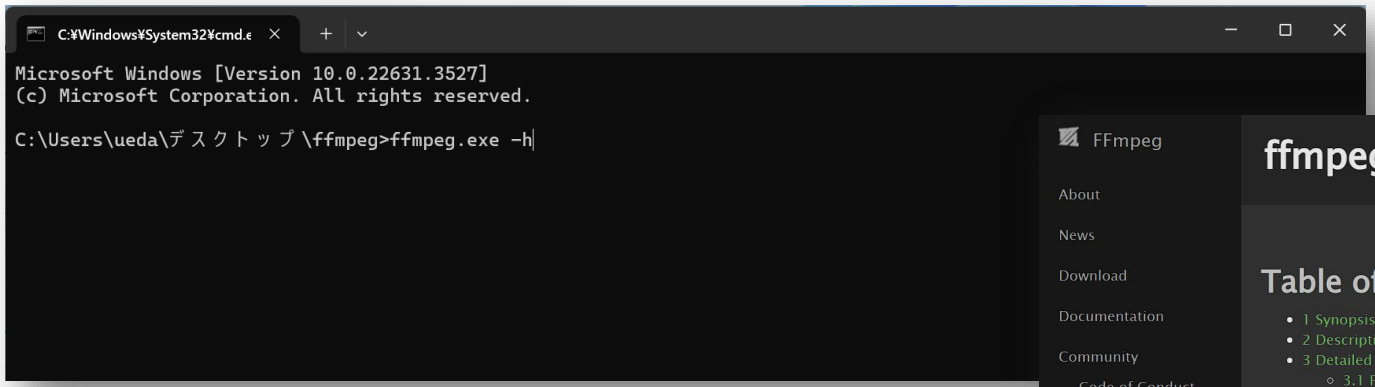
3.基本的な文法

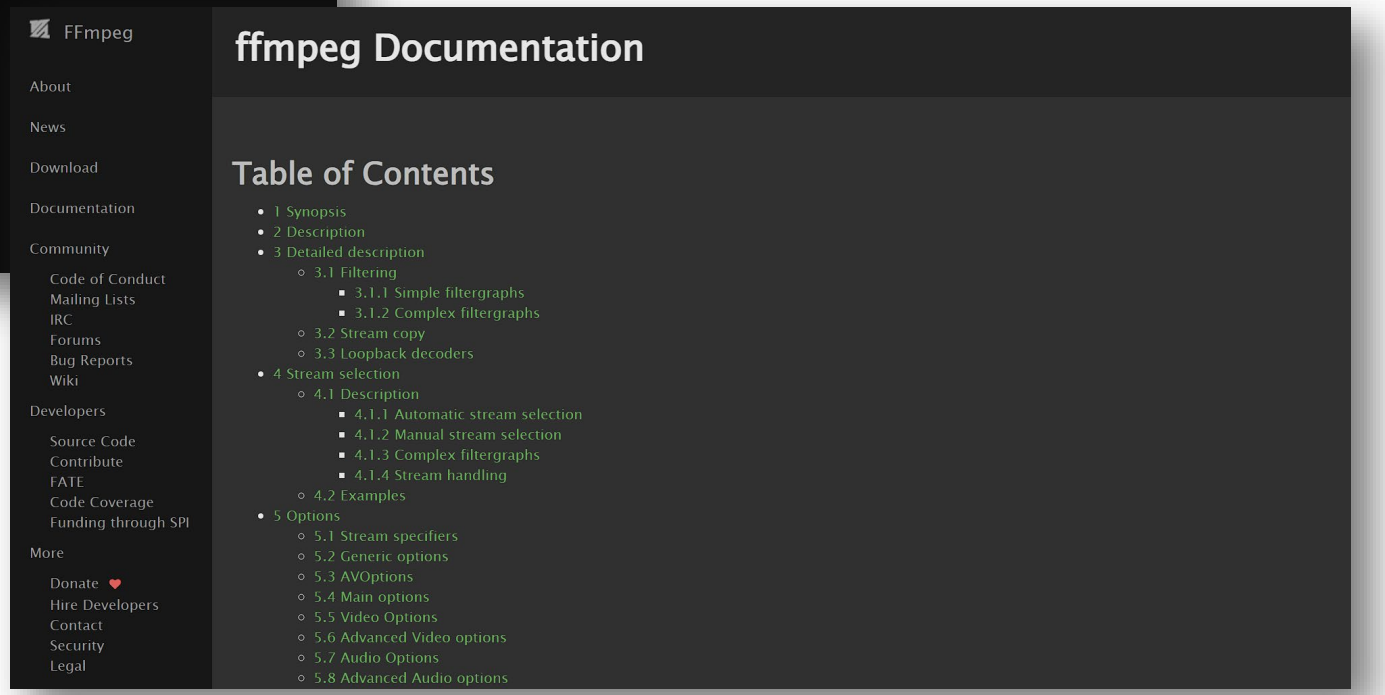
ORIHALCON Technologies

・オプションの調べ方

オプションやフィルターは膨大な数に上るので、ヘルプや公式ドキュメントを参考に

```
ffmpeg -h
```





https://ffmpeg.org/documentation.html

4.エンコードの基本

ORIHALCON Technologies

・そもそも「エンコード（符号化）」って？

映像や音声データを、特定の方法で復元可能な別の状態に変換する処理。対義語はデコード（エンコードされた情報を元に戻す復号の意）

掻い摘むと、制作されたマスターデータを映像再生ソフトウェアが扱いやすい形式に圧縮・変換することを「エンコード」と呼び、映像再生時には再生ソフトウェアが「デコード」処理を行い、映像を再生している。

インターネット等で配布されている天文可視化の映像データも、連番のままでは扱いにくいので何かしらでエンコードされている事が殆どである。（連番画像で配布される事も稀にあるが）

各ソフトウェアには推奨の映像形式があるので、再生するには適切にエンコードされている必要がある。拾ってきたままの映像を、再生できるからといってそのまま扱うのはあまりおススメしない。

4.エンコードの基本

ORIHALCON Technologies

```
ffmpeg -i INPUT.mov  –an –c:v mpeg4 –b:v 100M OUTPUT.mp4
```

-an　　出力ファイルにオーディオストリームを含めない

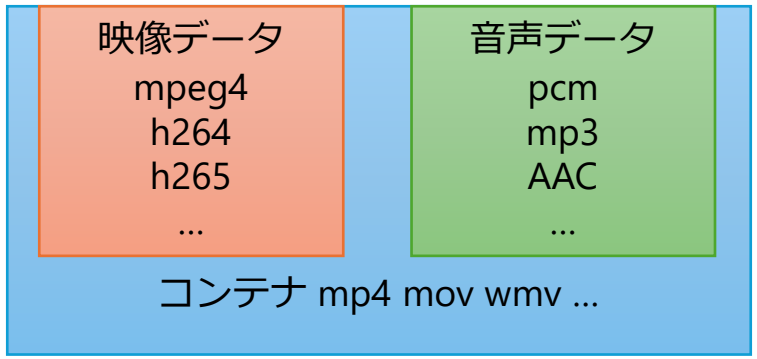
-c:v　　ビデオコーデックを指定する　この場合はmpeg4（エムペグフォー）というコーデックを指定している

-b:v　　出力ビットレートを指定する　他のオプションが無い場合は、指定したビットレートをターゲットにした可変ビットレートとしてエンコードされる

MEMO

コーデック：動画や音声を圧縮する際に使用されるアルゴリズムのこと。ひとくちに「mp4動画」といっても、使用されているコーデックが異なる場合は別物。「mp4」や「mov」といった、コーデックを格納するためのファイル形式のことは「コンテナ」と呼ぶ。

ビットレート：1秒間あたりのデータ量。単位はbps（Bit Per Second）。最終的な画質はビットレートと解像度、コーデックによって決まる。



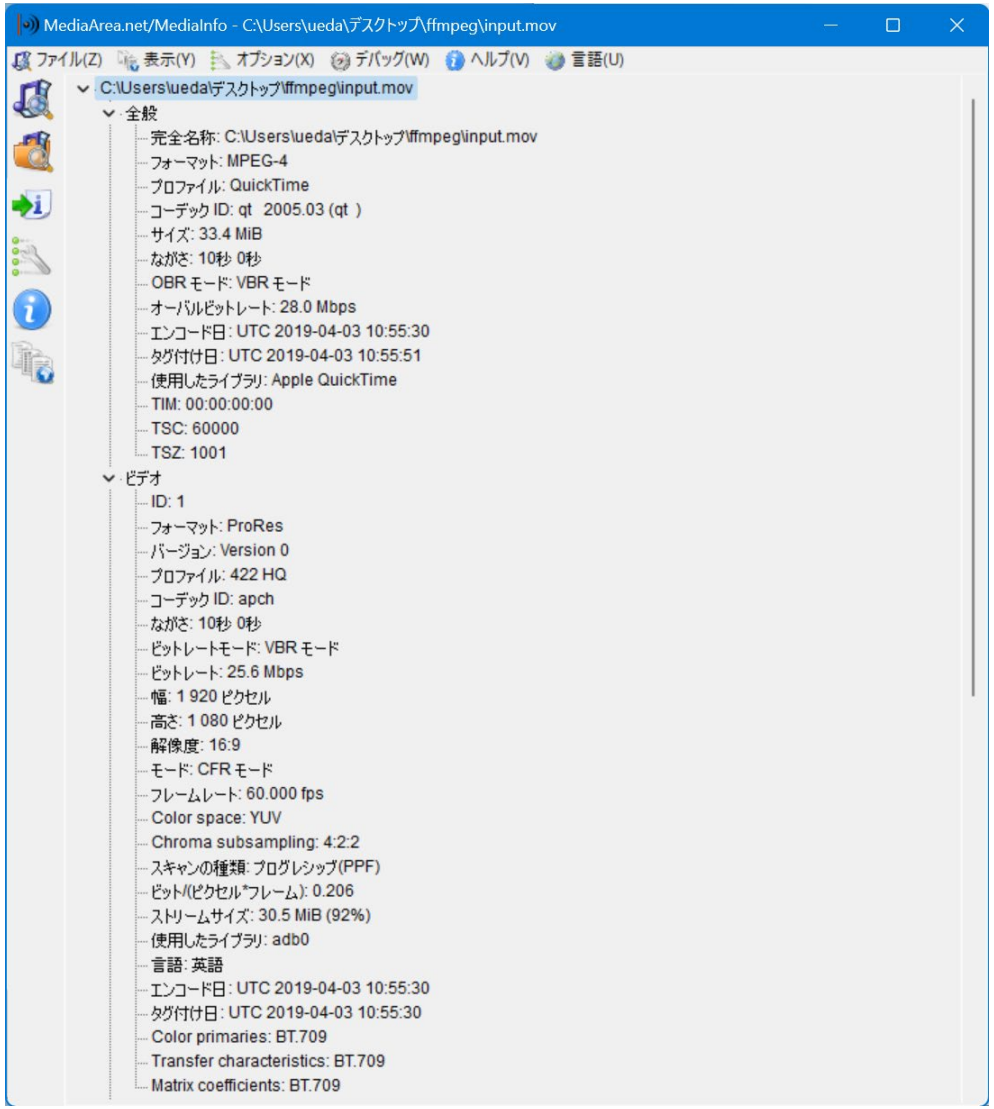
便利なツール紹介

ORIHALCON Technologies

- MediaInfo（メディアインフォ）

https://mediaarea.net/ja/MediaInfo

動画をドラッグ＆ドロップするだけで、コーデックや再生時間、解像度、ビットレートモードやフレームレートなど、様々な情報を一括で確認できる。



映像をリサイズする

ORIHALCON Technologies

```
ffmpeg -i INPUT –s 2048x2048 OUTPUT
```

-s WxH　　指定の解像度にリサイズする

動画の場合は再エンコード処理がかかるので、適切にビットレートをあててやる必要がある

-sで指定する値が入力映像のアスペクト比と異なる場合は絵が伸びたり縮んだりするので、4:3の映像の左右に黒をつけたい場合は後述のフィルター（-vf pad）を利用する

ドームマスター⇒エクイレクタングラー変換

```
ffmpeg -i INPUT -vf v360=fisheye:e:pitch=-90 OUTPUT
```

-vf ビデオフィルター　ここではv360というフィルタを利用している

ドームマスターをエクイレクタングラーに変換している

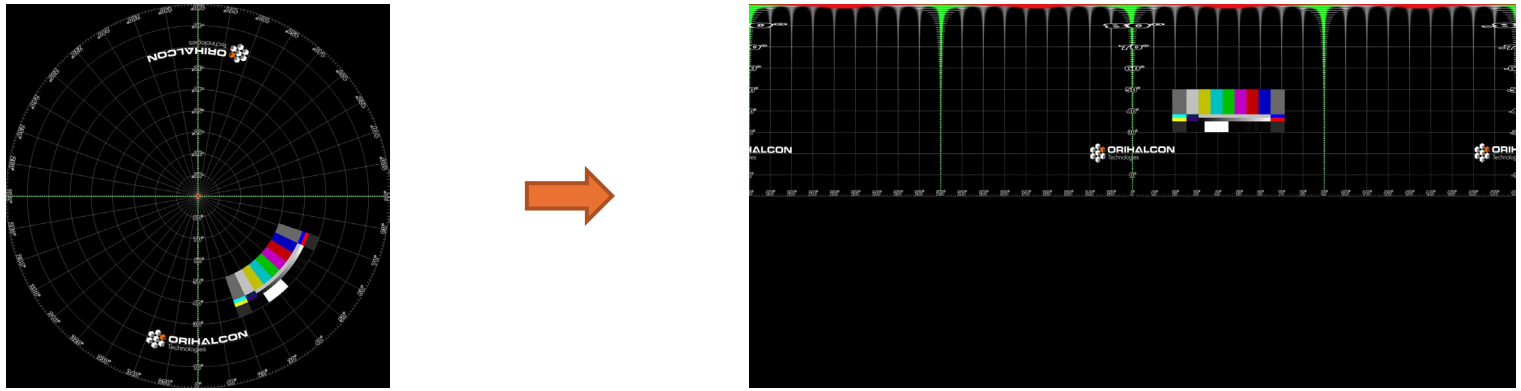
ドームマスターの1辺の解像度がエクイレクタングラーの縦解像度に出力される

正面を向くようにピッチを-90度回転

4K2Kエクイレクタングラーに書き出したい場合は、2Kドームマスターを入力するか、OUTPUTの直前に

-s 4096x2048を追加

-s 4096x2048　出力ファイルを幅4096px高さ2048pxにリサイズする



エクイレクタングラー⇒ドームマスター変換

ORIHALCON Technologies

```
ffmpeg -i INPUT -vf crop=4096:1024:0:0,pad=4096:2048:0:0,v360=e:fisheye:pitch=90 OUTPUT
```

-vf ビデオフィルター　ここではcropとpadとv360というフィルタを併用している。フィルタは前から順番に処理される。複数のフィルタを適用したい場合は、「，」で区切る

crop=W:H:x:y　　入力のx:y座標を起点に、W:Hにクロップする

pad=W:H:x:y　　入力にパディング（余白）を追加し、元の入力を指定されたx:y座標に配置する

4K2Kのエクイレクタングラーを INPUTとし、2Kドームマスターに変換している

上半分をクロップし、余白を黒で塗りつぶし、エクイレクタングラーをドームマスターに変換し、天頂を向くようにピッチを90度回転　エクイレクタングラーの縦解像度がドームマスターの一辺に相当する

4Kドームマスターに書き出したい場合は、8K4Kエクイレクタングラーを入力するか、OUTPUTの直前に

-s 4096x4096を追加。　※ 8K入力の場合はcropやpadの値も変わるので注意

1枚の静止画から指定秒の動画を作成

ORIHALCON Technologies

```
ffmpeg -loop 1 -i INPUT.bmp -c:v mpeg4 -t 3 -r 30 OUTPUT.mp4
```

プリビズ用にカット毎の動画を手早く作成したい、といった場合に

-loop 1　　入力を繰り返す

-t　　出力する秒数を指定する

-r　　出力するフレームレートを指定する

複数のカットをつなぐ

ORIHALCON Technologies

```
ffmpeg –f concat –i INPUT.txt –c copy OUTPUT.mp4
```

大量のカットを1本にまとめてプリビズを作る、といった場合に

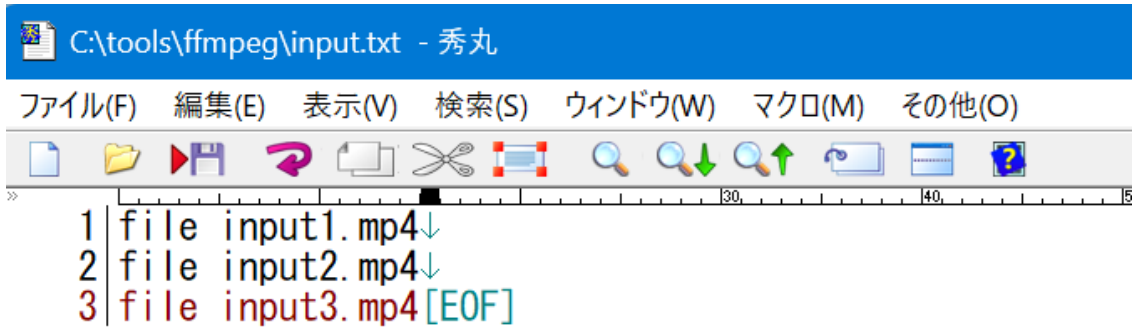
入力ファイルは別途テキストファイルに記述する　テキストには1行ずつ、ファイルパスを記入する

-f concat　入力ファイルのフォーマットにコンカット(結合)を強制する

-c copy　出力ファイルのコーデックは入力と同じものにする　再エンコードされないので処理が高速

※ 上記の場合、3つの入力ファイルが同じコーデックの場合、copyが使用可。それぞれ異なる場合は指定したコーデックで再エンコードされる。

※ 異なる解像度の動画同士は結合できない。その場合は先にリサイズ処理を行い、すべての動画について解像度を揃えておく必要がある。



トランジションをつける

ORIHALCON Technologies

```
ffmpeg -i INPUT1.mp4 -i INPUT2.mp4 -filter_complex xfade=transition=fade:duration=2:offset=5 –c:v mpeg4 OUTPUT.mp4
```

2カットをトランジションで繋げて1本の動画に　※ 2つの映像は同じ解像度、フレームレート、タイムベースでなければならない

-filter_complex　　ここではxfadeフィルターを指定している　※ 入力が複数の場合は-filter_complexを使う

xfade=以下は、xfadeフィルターのオプション指定

transition=fade:　　トランジションの種類を指定。ここでは単純なフェード。他にも多数種類がある。

※ 他のトランジションは　https://ffmpeg.org/ffmpeg-filters.html#xfade を参照

duration=2:　　トランジションが継続する時間を指定。デフォルトは1。

offset=5　　トランジションが開始する時間を指定。デフォルトは0。

上記の場合はINPUT1が5秒間表示され、2秒かけてINPUT2にフェードする。

MEMO

上記のように単純に2カットをトランジションでつなぐ程度であれば、わざわざ動画編集ソフトを使用するよりは遥かに簡単で早い。だからといってこれで長尺の動画全編を編集しようとするのはナンセンス。（やってやれなくはないだろうが、苦労が割に合わない。長尺でカットが多いなら素直に動画編集ソフトを使用するべき。）

連番画像を扱う

ORIHALCON Technologies

```
ffmpeg –framerate 30 –start_number 0 –i seq/seq_%5d.bmp -出力オプション OUTPUT
```

【入力編】タイムラプス動画の作成も可能

-framerate 30　入力する連番画像を30fpsであるとして扱う

-start_number　連番の開始番号を指定　この場合の入力は連番の0から開始される

-i seq/seq_%5d.bmp　seqフォルダ内の連番　%5dは5桁の意　seq_00000.bmpから始まる連番を入力

```
ffmpeg –i INPUT –f image2 –start_number 0 seq/OUTPUT_%5d.bmp
```

【出力編】この場合はINPUT動画を連番画像に切り出す

-f image2　画像に出力する

-start_number　連番の開始番号を指定　この場合の出力は連番の0から開始される

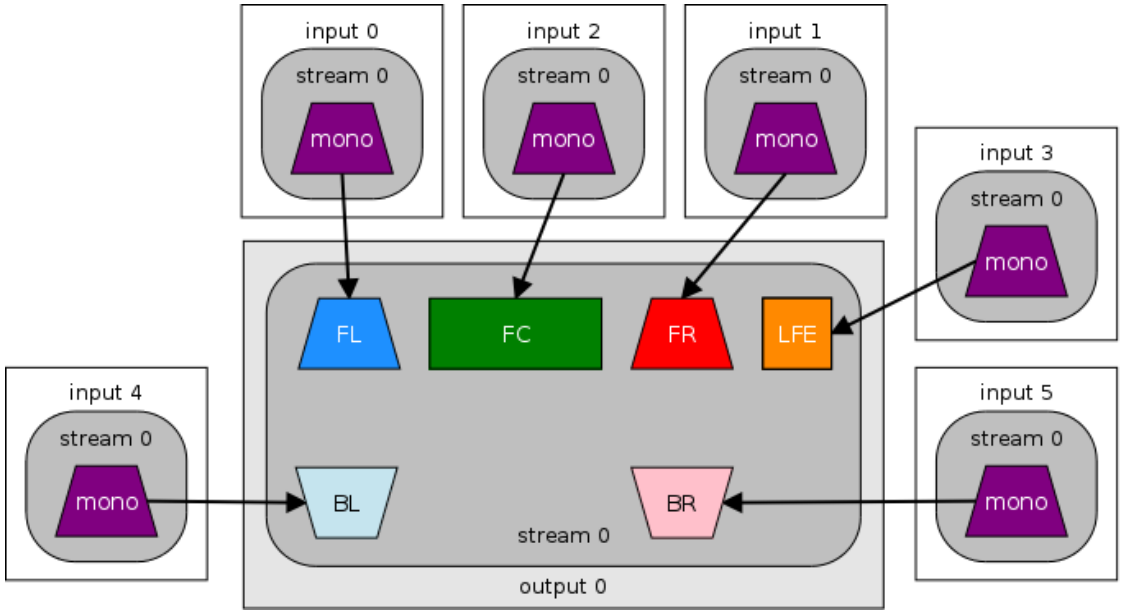
seqフォルダ内にOUTPUT_00000.bmpから始まる連番を出力

複数のモノラル音源を5.1ch音源にまとめる

ORIHALCON Technologies

```
ffmpeg -i FL.wav -i FR.wav -i C.wav -i LFE.wav -i BL.wav -i BR.wav -filter_complex "[0:a][1:a][2:a][3:a][4:a][5:a]amerge=inputs=6[a]" -map "[a]" -c:a copy output.wav
```

amerge	入力されたオーディオファイルをマージする
-map	オーディオのチャンネルマップを指定する
-c:a copy	オーディオコーデックは変更せず、入力ファイルと同じものにする 再エンコードされないので、処理が高速　品質も入力と変わらない



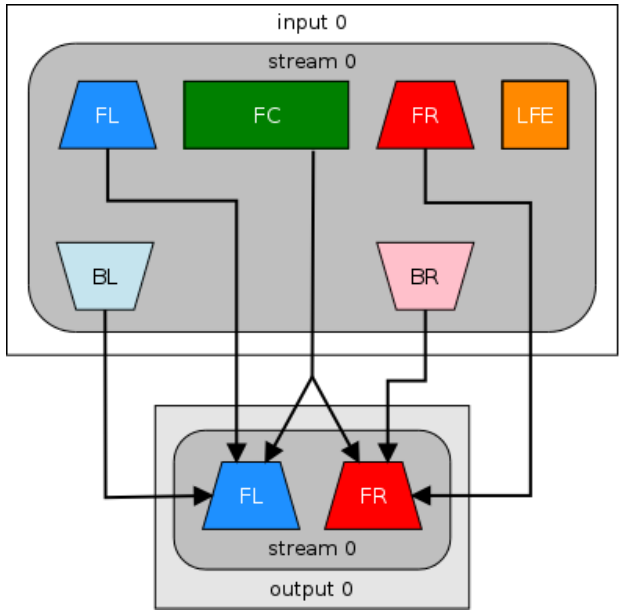
5.1ch音源をステレオ音源にダウンミックス

ORIHALCON Technologies

```
ffmpeg -i INPUT.wav –ac 2 OUTPUT.wav
```

-ac オーディオチャンネル数を設定する

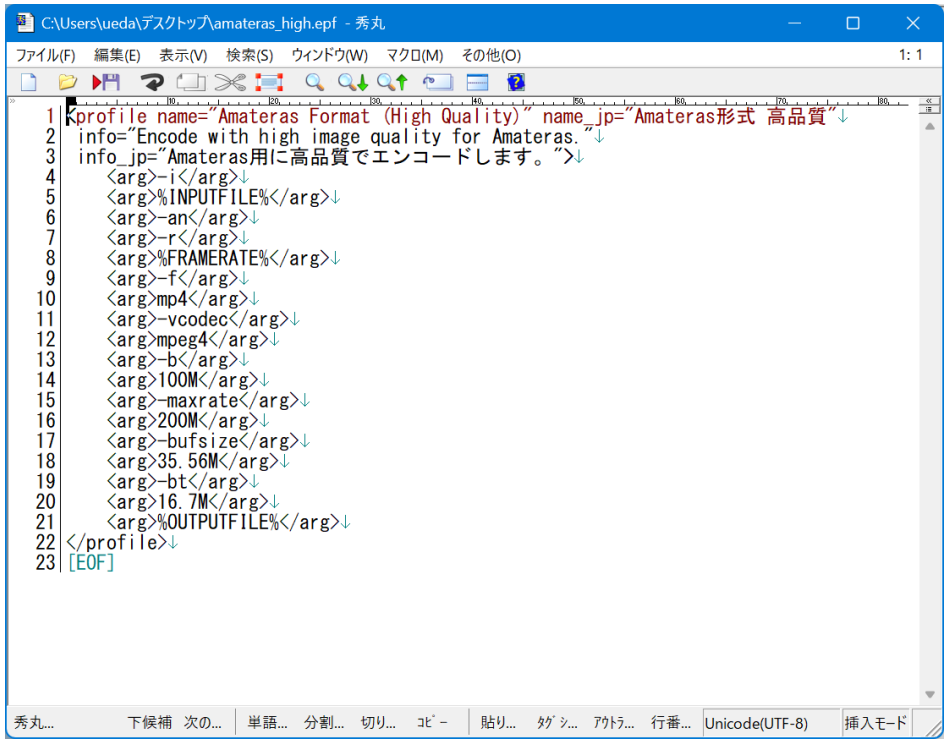
この方法でダウンミックスする場合、LFEの音声は無視される ※ 通常は問題にならないはず

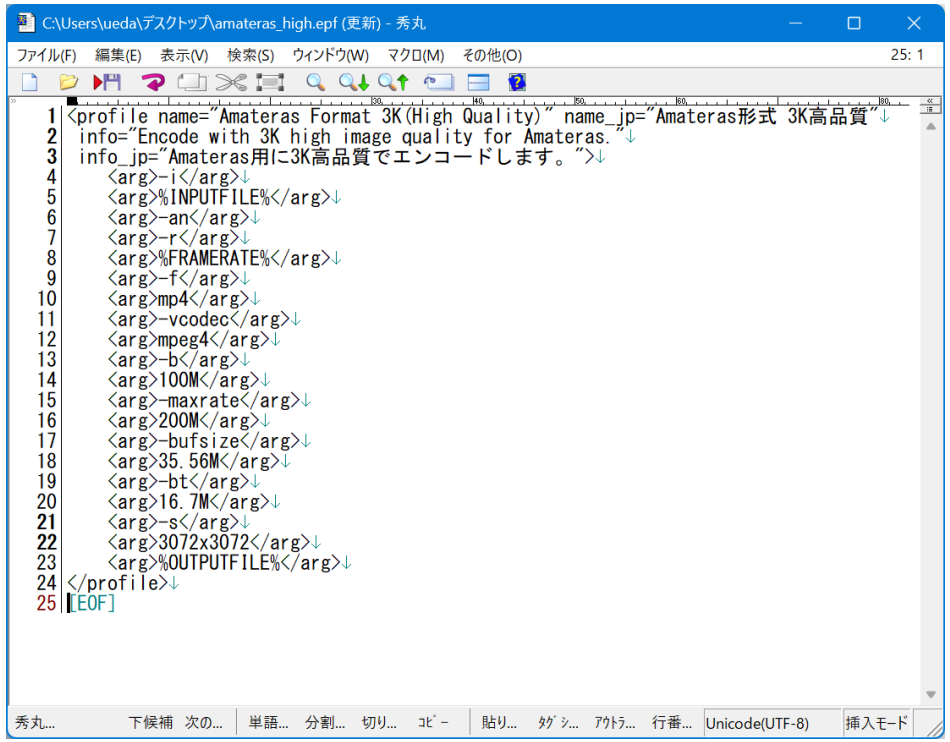


AmaterasEncoder用のプロファイルを改変する

AmaterasEncoderはFFmpegのフロントエンドとして動作しているので、プロファイルを編集することで独自にカスタムが可能、GUIからプロファイルに記載されている以外のパラメータを渡すこともできる

AmaterasDomeplayerの¥bin¥profiles内のepfファイルをコピー＆リネームしたら、右クリックしてテキストエディタで開いて編集　下記例では出力を3072pxにリサイズする処理を追加している





まとめ

ORIHALCON Technologies

FFmpegで出来ることで、特にドーム映像制作で役立ちそうな機能のごく一部を紹介しました。バッチファイルによる連続処理や定型処理も可能なので、一度覚えてしまえば便利に使えます。CUIによる操作には慣れが必要ですが、エンコードに失敗したところで元のファイルを破壊することはないので、気軽に試すことができます。
ここで紹介しきれないほど多くのフィルターが用意されているので、できることもまだまだ沢山あります。興味や必要があれば是非試してみてください。

※ 本文内でのスクリプト例では解説以外の出力オプションは省略しているところが多々あるため、実際に試してみる際には追記が必要な点には注意してください。

この内容を収めたPDFのダウンロードはこちら
https://www.orihalcon.co.jp/download/jpa2024_ffmpg.pdf