Kicect v2 를 이용하여 구현한 코드이므로 Kinect v1로 구현하려면 코드 수정이 필요함.

(수정 시, Body class -> Skeleton class 등 코드 전체적으로 수정이 필요할 것으로 보임.)

Kinect v2 를 이용한 개발환경:

* Kinect for windows SDK 2.0
* Kinect for windows developer toolkit v2.0.0 (설치 후 키넥트를 연결해주어야 함.)
* Microsoft .NET Framework 4

Skeleton 및 해당 좌표 구현 개발환경:

* Visual studio에서 WFP 앱(.NET Framework)로 프로젝트 생성
* Microsoft.Kinect.dll (2.0version) 참조 추가
* C# 언어로 개발
* Xaml 파일에서 control 을 이용하여 UI 구현

데이터 정제 및 패턴 분류를 위한 개발환경:

* Anaconda navigator(anaconda3)
* Jupyter notebook
* Python 언어로 개발
* 데이터는 skeleton에서 추출한 좌표값과 AI hub에서 받은 데이터를 이용(txt 형식)
* Anno\_3D.tar\_ 파일에 데이터를 정제하여 target\_data에 저장, 바탕화면에 두고 실행

먼저 visual studio에서 Extension.cs에서 전반적인 skeleton(body) 구현이 이루어짐.

* Camera 부분 (키넥트 촬영 부분) 구현, MainWindow.xaml에서 Camera로 송출
* 추가적인 Depth, Infrared 모드도 함께 구현됨
* Skeleton 구현의 Body 부분도 구현, Mainwindows.xaml에서 Canvas위에 그려짐.

카메라로 사람이 인식되면 먼저 관절 joint를 인식하고, 그 후에 skeleton 형식으로 연결.

MainWindow.xaml.cs에서 전반적인 실행 코드가 작성됨.

* 키넥트센서 활성화
* 측정이 되고 있음을 시각화하기 위한 타이머(시계) 설정
* 각 모드 실행
* Body 모드에서는 해당 사람의 joint별 좌표(x, y, z)를 저장(사람 구분, 최대 2명),

저장 경로는 현재 ‘D드라이브’로 설정

* MainWindow.xaml에서의 UI부분 코드 작성

Mainwindows.xaml 에서 UI 구성이 이루어짐(코드는 MainWindows.xaml.cs에서 작성됨)

* 각 모드 별 UI 구현
* START 버튼 클릭 시(클릭이벤트 설정), body 모드 실행 및 측정 안내 문구 실행
* STOP 버튼 클릭 시, body모드 종료 및 좌표 자동 저장 및 좌표 저장 안내 문구 실행,

Jupyter로의 연결; 그러나 데이터를 정제하여 코드를 따로 실행시켜야 함.

\*NeuralNetwork 폴더에 NN 관련 코드 저장

\*Capstone 폴더는 Kinect V1 버전의 코드

\*KinectStream 폴더를 user – source 로 복사하여 실행해야함.