MODE FOLDER

-RECEIVE\_MODE

\_\_init\_\_.py

-GUI에 수신 모드 디렉토리를 전달, 펼쳐진 폴더 버튼을 클릭하면, 파일을 여는 .py

-class ReceiveMode(파일을 받기위한 윈도우 UI의 한 부분)

Def init

Web object 생성 -> 서버 상태 설정 -> 서버에 업데이트하라고 요청 -> 기록 업로드

Def get\_stop\_server\_shutdown\_timeout\_text(self)

만약에 shutdown되야할 시간이 넘으면, String 리턴, 서버 멈춤버튼으로

Def timeout\_finished\_should\_stop\_server

Shutdown timer가 없어졌으면, 서버를 꺼야하냐고 bool값으로 값을 줌.

Def start\_server\_custom

서버 시작.

Def start\_sever\_step2\_custom

서버를 시작하기위한 두번째 단계, 세번째 단계로 향하는 함수가 안에 있음.

self.starting\_server\_step3.emit()

def handle\_tor\_broke\_custom

tor\_broke에 연결함.

Def handle\_request\_load

Request\_load를 제어함.

Def handle\_request\_started

Request\_started를 제어.

History를 불러오고, history를 업데이트함.

Def handle\_request\_progress

request\_progress를 제어

history를 업데이트함. Progress로

def handle\_request\_close\_server

request\_close\_server를 제어

서버를 멈추고, 메시지를 띄움.

def handle\_request\_upload\_file\_renamed

request\_upload\_file\_renamed를 제어

history업데이트, 이름을 바꿈.

def handle\_request\_upload\_set\_dir

request\_upload\_set\_dir를 제어

디렉토리를 설정함.

def handle\_request\_upload\_finished

request\_upload\_finished를 제어

history를 끝났다고 업데이트함.

def on\_reload\_settings

Start Receive Mode가 다시 사용가능하다고 알림.

def reset\_info\_counters(self):

history를 리셋함.

def update\_primary\_action(self)

중요한 작업을 띄움.

SHARE MODE

파일 sharing 윈도우 UI에 한 부분.

\_\_init\_\_.py

def init

쓰레드를 시작함->web object생성->파일 선택 -> 서버상태체크->파일사이즈 waring->history 다운 ->label 붙임->history toggle ->

def get\_stop\_server\_shutdown\_timeout\_text

server를 멈추기위해 string을 반환.

def timeout\_finished\_should\_stop\_server

shutdown 시간이 지나면, should we stop server라는 문자열을 출력하고 bool을 리턴함.

def start\_server\_custom

서버를 시작함.

def start\_server\_step2\_custom

서버를 시작하기위한 2번째 스텝, 파일을 압축함.

def start\_server\_step3\_custom

서버를 시작하기 위한 3번째 스텝, 파일 압축의 정도를 확인하는 바를 지우고, 파일 사이즈가 넘는다면 알림을 띄움

def start\_server\_error\_custom

서버를 시작할 때의 에러를 알려줌.

def stop\_server\_custom

서버를 멈춤.

def cancel\_server\_custom

압축 스레드를 중지시킴.

def handle\_tor\_broke\_custom

tor broke에 연결.

def handle\_request\_load

요청들을 로드함.

def handle\_request\_started

시작 요청을 처리함

def handle\_request\_progress

다운로드가 끝났는지 확인하고 닫을지 확인함.

def handle\_request\_canceled

progress의 count를 업데이트해줌.

def on\_reload\_settings

공유된 파일이 있다면, start Sharing 버튼을 활성화 시킴.

def update\_primary\_action

라벨에 있는 파일의 개수를 업데이트함.

def reset\_info\_counters

count를 0으로 초기화함.

def \_compute\_total\_size

total\_size를 측정함.

class class ZipProgressBar

압축한 과정들의 사이즈와 신호들을 기록함.

File\_selection.py

class DropHereLabel(파일이나, 폴더라 리스트에 아직 없다면, drop files here라는 메시지와 그래프를 표시함)

def \_\_init\_\_

def dragEnterEvent

class DropCountLabel(파일리스트에 파일들을 드래그하면, 이 측정기가 드래그한 파일의 개수를 보여줌.)

def \_\_init\_\_

def dragEnterEvent

class FileList(GUI에 파일 리스트와 폴더를 표시)

def \_\_init\_\_

def update

GUI의 요소들을 현재상태에 기초하여 업데이트함. 파일리스트를 사용하려먼 백그라운드에 이미지가 있어야함.

def server\_started

서버가 시작할 때, delete버튼을 숨김으로써 GUI를 업데이트해줌.

def server\_stopped(self):

서버가 정지했을 때, delete버튼을 표시함으로써 GUI를 업데이트해줌.

def resizeEvent(self, event)

위젯을 resize했을 때, 드롭파일의 이미지와 텍스트를 resize해줌.

def dragEnterEvent (self, event)

위젯 안으로 파일과 디렉토리를 드래그 했을 때, 이벤트발생. 위젯에 정보를 작성함.

def dragLeaveEvent(self, event):

위젯 밖으로 파일과 디렉토리를 드래그 했을 때, 이벤트발생. 위젯에 올라온 정보들을 삭제함.

def dragMoveEvent(self, event)

위젯 밖으로 파일과 디렉토리를 드래그 했을 때, 이벤트발생. 드래그를 위한 함수.

def dropEvent(self, event):

파일이름을 재정의하고, setDropAction을 부름.

def add\_file(self, filename):

위젯에 파일 또는 디렉토리를 추가함.

def delete\_item(): // 이게바로 delete 버튼임.

Delete 버튼을 만듦.

class FileSelection(QtWidgets.QVBoxLayout): (파일과 폴더의 리스트들이 GUI안에 있을 때, 추가와 삭제 버튼을 만듦)

def \_\_init\_\_(self, common):

class를 시작하는 함수.

def update(self):

현재 상태에 맞게 GUI를 업데이트함.

Def add(self):

추가 버튼이 클릭되었을 때, 실행하는 함수. 파일 리스트에 파일을 추가하고, 업데이트해줌.

def delete(self):

삭제 버튼이 클릭되었을 때, 실행하는 함수. 파일리스트에 파일을 삭제하고, 업데이트해줌.

def server\_stopped(self):

서버가 정지되었을 때, sever의 상태를 변경하고 업데이트.

def get\_num\_files(self):

리스트에 있는 총 파일의 개수를 리턴해줌.

def setFocus(self):

파일 선택 박스에 앱을 세팅함.

Threads.py

class CompressThread(QtCore.QThread)(공유하기위해 파일을 압축함.)

def \_\_init\_\_(self, mode):

클래스 시작함수.

def set\_processed\_size(self, x):

압축중인 파일의 진행상황을 bar로 보여줌.

def run(self):

thread가 시작됨을 알리고 시작.

def cancel(self):

웹과 파일압축프로그램에게 취소되었다는 사실을 알림.

\_\_init\_\_.py

class Mode(QtWidgets.QWidget):(share모드와 receive모드를 상속시킴. )

def \_\_init\_\_(self, common, qtapp, app, status\_bar, server\_status\_label, system\_tray, filenames=None, local\_only=False):

클래스 시작.

def init(self)

사용자 재설정이 필요하다면 여기다가 하면 됨.

def timer\_callback(self):

타이머에서 일반적으로 사용되는 메소드. Auto shutdown timer가 멈추면, 서버를 멈춤.

서버 버튼을 업데이트함.

def timer\_callback\_custom(self):

timer 사용자 설정.

def get\_stop\_server\_shutdown\_timeout\_text(self):

shutdown이 되었다면, 서버 멈춤 버튼에 string을 리턴,

def timeout\_finished\_should\_stop\_server(self):

shutdown timer가 제 기능을 못하면, should we stop the server를 출력하고, bool을 리턴.

def start\_server(self):

onionshare server를 시작하고, 웹과 tor onionserver를 멀티쓰레드로 동시에 돌림.

def start\_server\_custom(self):

사용자 설정.

def start\_server\_step2(self):

onionshare server를 키기위한 두번째 스텝

딱히쓴 건 없는 거 같고, log를 써놓음.

def start\_server\_step2\_custom(self):

사용자 설정.

def start\_server\_step3(self):

3번째 스텝, shutdown time을 정하고, 유저가 이 버튼을 클릭하기전에 미리 실행됨. 서버실행의 시간이 오래걸리기 때문.

def start\_server\_step3\_custom(self):

사용자 설정.

def start\_server\_error(self, error):

onionshare를 시작할 때 에러가 있으면 잡아줌.

def start\_server\_error\_custom(self):

사용자 설정.

def cancel\_server(self):

시작을 준비할 때, 서버를 cancel시킴.

def cancel\_server\_custom(self):

사용자설정

def stop\_server(self):

onionshare 서버를 정지시킴.

def stop\_server\_custom(self):

사용자설정

def handle\_tor\_broke(self):

tor breaking에 연결을 관리함.

def handle\_tor\_broke\_custom(self):

사용자설정

|  |  |
| --- | --- |
|  | Webserver에 이벤트들을 처리하는 함수. |
|  | def handle\_request\_load(self, event): |
|  | """ |
|  | Handle REQUEST\_LOAD event. |
|  | """ |
|  | pass |
|  |  |
|  | def handle\_request\_started(self, event): |
|  | """ |
|  | Handle REQUEST\_STARTED event. |
|  | """ |
|  | pass |
|  |  |
|  | def handle\_request\_rate\_limit(self, event): |
|  | """ |
|  | Handle REQUEST\_RATE\_LIMIT event. |
|  | """ |
|  | self.stop\_server() |
|  | Alert(self.common, strings.\_('error\_rate\_limit'), QtWidgets.QMessageBox.Critical) |
|  |  |
|  | def handle\_request\_progress(self, event): |
|  | """ |
|  | Handle REQUEST\_PROGRESS event. |
|  | """ |
|  | pass |
|  |  |
|  | def handle\_request\_canceled(self, event): |
|  | """ |
|  | Handle REQUEST\_CANCELED event. |
|  | """ |
|  | pass |
|  |  |
|  | def handle\_request\_close\_server(self, event): |
|  | """ |
|  | Handle REQUEST\_CLOSE\_SERVER event. |
|  | """ |
|  | pass |
|  |  |
|  | def handle\_request\_upload\_file\_renamed(self, event): |
|  | """ |
|  | Handle REQUEST\_UPLOAD\_FILE\_RENAMED event. |
|  | """ |
|  | pass |
|  |  |
|  | def handle\_request\_upload\_set\_dir(self, event): |
|  | """ |
|  | Handle REQUEST\_UPLOAD\_SET\_DIR event. |
|  | """ |
|  | pass |
|  |  |
|  | def handle\_request\_upload\_finished(self, event): |
|  | """ |
|  | Handle REQUEST\_UPLOAD\_FINISHED event. |
|  | """ |
|  | Pass |

History.py

class HistoryItem(QtWidgets.QWidget)기본적인 history item

def \_\_init\_\_(self):, def update(self):, def cancel(self):

class DownloadHistoryItem(HistoryItem): sharemode를 위한 history item download

def \_\_init\_\_(self, common, id, total\_bytes):

함수 시작.

def update(self, downloaded\_bytes):

history update

def cancel(self):

gui에 그냥 cancel버튼만 만듦.

def estimated\_time\_remaining(self):

기억된 시간을 return 해줌.

class UploadHistoryItemFile(QtWidgets.QWidget): history item file을 업로드

def \_\_init\_\_(self, common, filename):

라벨과 버튼들을 추가함.

def update(self, uploaded\_bytes, complete):

라벨사이즈를 보여주고 출력.

def rename(self, new\_filename):

이름바꾸기

def set\_dir(self, dir):

dir설정함.

def open\_folder(self):

다운로드 폴더에서 선택된 파일들을 열음. Mac, linux, window가 다 따로 있음.

class UploadHistoryItem(HistoryItem):

def \_\_init\_\_(self, common, id, content\_length):

라벨, 바, 레이아웃들을 설정함.

def update(self, data):

각 파일의 파일 사이즈 라벨들을 업데이트함. Runtime error를 피하기 위해 사용한 코드도 있음.

class HistoryItemList(QtWidgets.QScrollArea): 아이템 리스트들

def \_\_init\_\_(self, common):

아이템들의 레이아웃과 스크롤들을 구현.

def resizeScroll(self, minimum, maximum):

크기가 변경될 때마다 창의 마지막으로 스크롤함.

def add(self, id, item):

새로운 아이템을 만들고, overridie 해줌.

def update(self, id, data):

아이템들을 업데이트.

def cancel(self, id):

아이템들을 cancel

def reset(self):

아이템들을 리셋.

class History(QtWidgets.QWidget): 각각의 모드에서 무슨일이 일어났나 history를 가지고있음. 스크롤이 가능한 아이템 리스트를 가지고있음.

def \_\_init\_\_(self, common, empty\_image, empty\_text, header\_text):

레이아웃 제작.

def add(self, id, item):, def update(self, id, data):, def cancel(self, id):, def reset(self):, def update\_completed(self):, def update\_in\_progress(self):

class ToggleHistory(QtWidgets.QPushButton):

history들을 숨기거나 볼 수 있음.

def \_\_init\_\_(self, common, current\_mode, history\_widget, icon, selected\_icon):

레이아웃 생성.

def update\_indicator(self, increment=False):

count를 증가시켜주고, 업데이트하여 보여줌.

def toggle\_clicked(self):

history를 보여주거나 숨김

RECEIVE\_MODE FOLDER

Uploads.py

class File(QtWidgets.QWidget):

def \_\_init\_\_(self, common, filename):

레이아웃 생성.

def update(self, uploaded\_bytes, complete):, def set\_dir(self, dir):, def rename(self, new\_filename):, def open\_folder(self):(Linux, mac,Windows),

class Upload(QtWidgets.QWidget):

def \_\_init\_\_(self, common, upload\_id, content\_length):

레이아웃 생성.

def update(self, progress):, def rename(self, old\_filename, new\_filename):, def set\_dir(self, filename, dir):

def finished(self):

progress bar를 숨기고 라벨을 바꿈.

class Uploads(QtWidgets.QScrollArea): GUI에 상당량에 파일들을 업로드함.

def \_\_init\_\_(self, common):

레이아웃 생성

버튼 액션:

def resizeScroll(self, minimum, maximum):, def add(self, upload\_id, content\_length):, def update(self, upload\_id, progress):, def rename(self, upload\_id, old\_filename, new\_filename):, def finished(self, upload\_id):, def cancel(self, upload\_id):, def reset(self):

def resizeEvent(self, event):

Share\_mode FOLDER

Downloads.py

class Download(object):

def \_\_init\_\_(self, common, download\_id, total\_bytes):

def update(self, downloaded\_bytes):

def cancel(self):

def estimated\_time\_remaining(self):

class Downloads(QtWidgets.QScrollArea):

def \_\_init\_\_(self, common):

레이아웃 생성

def resizeScroll(self, minimum, maximum):

창의 변경된 범위를 바꿔줌.

def add(self, download\_id, total\_bytes):

새로운 다운로드 진행 바를 추가함.

def update(self, download\_id, downloaded\_bytes):

다운로드 진행 바를 업데이트

def cancel(self, download\_id):

다운로드 진행바가 cancel된 것을 보여줌

def reset(self):

다운로드 진행바를 0으로 초기화함.