# INI Intern 1st Project Report

## 목차



## 1. 담당 프로그램 정의

● 프로그램 제작 기간
2018.07.02 ~ 2018.07.22 (20 일)

● 프로그램 목적.
- 페이지에서 동영상이 업로드 될 때, 프로그레스바를 띄어 동영상 업로드 과정을 보여주며 업로드의 성공 유무를 출력.
- 업로드 할 동영상을 사용자에 맞게 지정된 경로로 이동

## 2. 개발 환경

● 서버
○ubuntu(16.0.4 version)

● 프론트엔드
○Framework : Wordpress Snaptube

● 백엔드
○ Framework : PHP, Python(3.5.2)
○ Editor : vi

# 3. Wordpress 구현

## ● 업로드 버튼에 의한 'uploadFile()' 함수 호출

- ajax 를 이용한 request 요청을 준비
- 단, 서버와 통신이 잘되지 않거나 동영상에 결점이 있다면 EventListener 를 통해 오류 처리.

```
<form method="post"  enctype="multipart/form-data">
<input type="file" name="file1" id="file1"><br>
<input type="button" value="Upload File" onclick="uploadFile()">
<progress id="progressBar" value="0" max="100" style="width:300px;"></progress>
<h3 id="status"></h3>
<p id="loaded_n_total"></p>
```

```
var formdata = new FormData();
formdata.append("file1", file);
var ajax = new XMLHttpRequest();
ajax.upload.addEventListener("progress", progressHandler, false);
ajax.addEventListener("load", completeHandler, false);
ajax.addEventListener("error", errorHandler, false);
ajax.addEventListener("abort", abortHandler, false);
```

## ● progressHandler() 함수

- USER 가 upload 버튼을 클릭했을 때, 프로그레스바가 작동
- Math.round 함수를 구현, 프로그레스바의 실행 흐름을 유관으로 확인 가능
- 동영상 업로드가 완료 될 때 까지 "please wait"를 출력

```
function progressHandler(event)
{
 _("loaded_n_total").innerHTML = "Uploaded "+event.loaded+" bytes of "+event.total;
 var percent = (event.loaded / event.total) * 100;
 _("progressBar").value = Math.round(percent);
 _("status").innerHTML = Math.round(percent)+"% uploaded... please wait";
}
```

- 프로그레스바의 진행이 완료 된 후, PHP 또는 Python APISTAR 에 의해 파일이 지정한 경로로 갔는지 확인 가능

## ● Ajax 의 "POST"요청 → PHP, Pyhon APISTAR 작동

○ API

URL /post_sentence
Method: POST
Data Params : (username = +userName), send(formdata)

○ Success Response

Code : 200

○ Fail Response

Code : 404
Content : Not Found

Code :500
Content : Internal Server Error

PHP

```
ajax.open("POST", "http://localhost/file_upload_parser.php?user_name="+userName);
ajax.send(formdata);
```

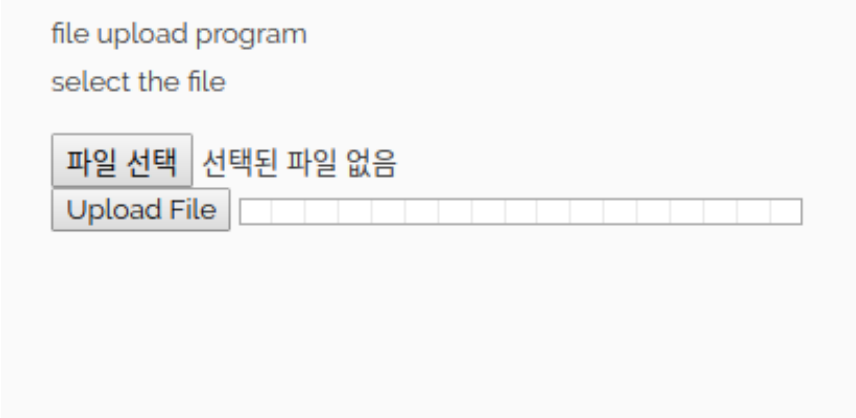
Python APISTAR

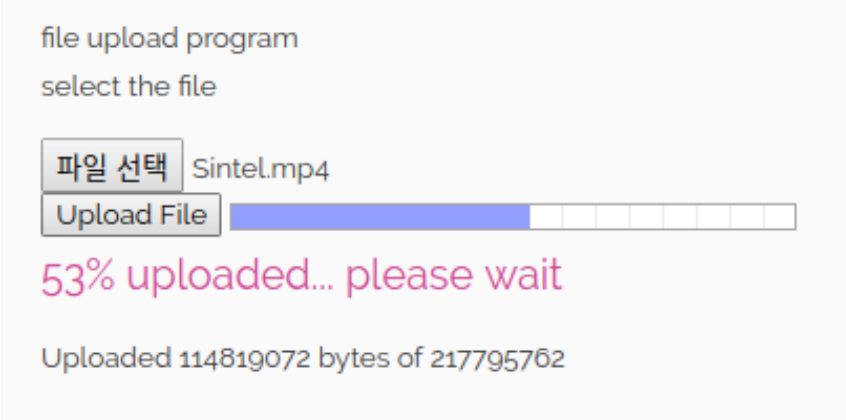
```
ajax.open("POST", "http://127.0.0.1:5000/upload");
ajax.send(formdata);
```

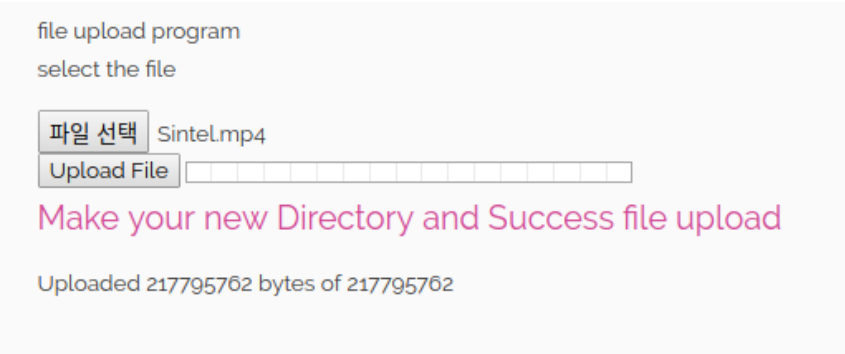
* **한계** : 사용자 이름('username')을 가지고 사용자를 구분하는 것은 보안 등의 문제로 좋은 방법이 아님.

→ 로그인 된 사용자의 정보를 알 수 있는 쿠키를 이용하는 것이 더 좋을 것으로 보임.

## ● 실행 화면






# 4. PHP 로 구현한 프로그레스바

## ● 실행 흐름

1) wordpress 에서 “POST” 요청과 함께 업로드 할 데이터 전송, 이 때 사용자 구분을 위해 사용자의 이름을 같이 전송.

2) file_upload.php 에서 전송된 데이터를 받고 똑같은 이름의 파일이 이미 경로에 있는지 검사 후에 동영상 업로드.

3) 단, 똑같은 이름의 파일이 존재한다면 에러 메세지 출력 후 실행 종료.

## ● 구현

Wordpress 에서 전송 된 formdata 를 받아 각각 변수로 지정
$_GET()을 통해 별도로 전송된 username 을 지정

```
$fileName = $_FILES["file1"]["name"];
$fileTmpLoc = $_FILES["file1"]["tmp_name"];
$fileType = $_FILES["file1"]["type"];
$fileSize = $_FILES["file1"]["size"];
$fileErrorMsg = $_FILES["file1"]["error"];
$userName = $_GET["user_name"];
$fileDes = "/var/www/html/wp-content/uploads/$userName";
$uploadfile = "$fileDes/$fileName";
```

업로드 될 경로가 있는지 확인 가능

```
is_dir($fileDes)
```

업로드 될 경로가 없다면 경로를 생성

```
mkdir($fileDes)
```

업로드 할 동영상을 지정한 경로로 이동

```
move_uploaded_file($fileTmpLoc, "$fileDes/$fileName" )
```

# 5. Python APISTAR 로 구현한 프로그레스바

## ● 실행흐름

PHP 로 구현한 프로그레스바 소스를 Python APISTAR 를 이용하여 변경.

1) Wordpress 의 “POST”요청을 APISTAR 가 작동 중인 URL 로 전송
→ APISTAR 는 Wordpress 의 POST 요청을 받아 작동

2) 경로의 유무를 확인하고 파일을 저장하거나 에러 메세지를 출력
3) 함수 실행이 끝난 후에 요청 완료 Respose 를 보냄

## ● 모듈

○ import http, import requests : 서버의 요청을 받을 수 있는 기능을 지원하는 모듈

```
from apistar import http
import requests
```

○ import Path, import os : 업로드 파일의 경로를 파악하고 지정 할 수 있는 기능을 지원하는 모듈

```
from pathlib import Path
import os
```

○ import App, Route : 웹 페이지에서 콘텐츠를 볼 수 있는 기능을 지원하는 모듈 → upload_file 함수를 호출 하여 동영상 업로드 및 경로 이동

```
from apistar import App, Route
```

```
routes = [
 Route('/upload', method='POST', handler=upload_file),
]

app = App(routes=routes)
```

## ● 구현

http.RequestData 를 이용하여 “POST”로 넘어온 formdata 를 받고 file 에 저장

```
file = request_data.get('file1')
```

업로드 될 파일이 저장 될 경로가 존재하는지 확인

```
os.path.isdir(fileDes):
```

경로를 만들어줌

```
os.makedirs(str(fileDes))
```

같은 경로에 똑같은 이름의 파일이 존재하는지 확인

```
os.path.isfile( str(fileDes)+'/'+filename ):
```

파일 저장

```
file.save(os.path.join(fileDes,filename))
```

서버로 업로드 함수가 끝났음을 알리는 메세지를 보냄

```
content =  'Success file upload'
headers = {'Content-Type' : 'text/plain'}
return http.Response(content, headers=headers)
```

# 6. 애로 사항 및 발전 요구 사항

## ● 애로 사항

- Wordpress 에서 데이터를 전송할 파일의 이름을 'file1'으로 작성. file 의 이름을 계속 "file"로 적어 에러 발생. 따라서 Wordpress 에서 넘어오는 데이터 포맷을 신경 쓸 것.

- 예를 들어 "/var/www/html/wp-content/uploasds"를 변수에 저장하지 않고 단독으로 사용하면 PHP 에서 인식하지 못해서 에러 발생.  따라서 변수에 경로에 해당하는 값을 넣고 사용 해야함.

- APISTAR 로 구현 할 때 계속 404 ERROR 가 발생. "POST"에만 초점을 맞춰 에러를 해결하려 했으나 하지 못함. 정답은 Route('/upload', method='OPTIONS', handler=function_name) 로 method 가 'OPTIONS' 일 때의 처리도 필요.

## ● 발전 요구 사항

- Wordpress 에서 username 을 가지고 사용자를 구분하는 데는 보안 등의 문제로 좋은 방법이 아님.
PHP 와 APISTAR 에서 로그인 된 사용자의 정보를 알 수 있는 쿠키를 이용하는 것이 더 좋을 것으로 보임.

- 동영상에 오류가 존재하거나 업로드 될 파일의 경로에 이미 같은 이름의 파일의 유무를 확인하게 하는 코드로 구현하여 프로그레스바가 진행되기 전에 확인하게 하는 방법으로 구현하는 것이 더 좋을 것으로 보임.

# 7. 후기

-  처음에 Wordpress 가 제공하는 플러그인 및 기능들을 숙지하는 것 이미 짜여진 틀 안에서 개발을 하는 것에 많은 어려움이 있었다. 언어를 다루는 데에도 많은 시행착오가 존재했다. Wordpress 에서 제공하는 PHP 플러그인은 페이지 편집 내 자바스크립트를 적는 텍스트에 [insert_php]를 사용하여 PHP 코드를 입힐 수 있었다. PHP 를 다루어 본 적이 없었기에 언어를 숙지하는 것부터 시작해서 Wordpress 에 적용을 하고 코드를 짜는 것은 매우 어려운 일이었다. 하지만 Wordpress 로 인해 숙달 된 만큼 업로드 할 파일의 경로를 설정하고 동영상을 업데이트하는 코드를 짜는 것은 짧은 시간 내에 해결 할 수 있었다.
 그 후에 PHP 코드로 작성한 파일을 Python APISTAR 로 수정하는 과제는 더욱더 어려웠다. PHP 와 마찬가지로 Python 을 다루어 본 적이 없었고 무엇보다 APISTAR 에 대한 정보가 구글에 많이 나와있지 않았기 때문이다. 따라서 우선 APISTAR 보다 정보가 많은 Flask 로 코드를 작성했다. Flask 는 다양한 모듈을 제공해서 개발자가 편리하게 사용할 수 있었다. 또한 코드를 작성하며 빠르게 파이썬 문법을 숙달 할 수 있었다. Flask 로 코드를 구현하는 것에는 많은 시간이 걸리지 않았다. Flask 를 구현해본 덕에 API 로 구현하는 것은 어렵지 않았다. PHP 코드를 작성하며 공부했던 GET, POST 등을 다시 한번 공부 할 수 있었고 서버에 대하여 더욱더 자세히 알 수 있었다.