

학기말 프로젝트

'맘 속의 수 맞추기'

학과 : 소프트웨어학부

학번 : 20181630

이름 : 송경민

강의:소프트웨어 프로젝트

서버의 내부 설계

1. 핵심 논리 모듈 개발

```
picknumber.py (~/final/final-proj) - gedit
열기(O)  [icon]

import random

class Number:

    def __init__(self):
        self.secret = 0
        self.digit = 0
        self.trials = 0

    def newGame(self, amount):
        self.secret = random.randint(1, amount)
        self.digit = amount
        self.trials = 0

    def guess(self, userGuess):
        self.trials += 1
        if 0 < userGuess < self.secret:
            return "You have to type bigger number."
        elif userGuess > self.secret:
            return "You have to type smaller number."
        elif userGuess == 0:
            return "0 is out of range"
        elif userGuess == self.secret:
            return "FINISH"

    def getGuessCount(self):
        return self.trials

if __name__ == '__main__':
    s = Number()
    count = int(input("Set Range : "))
    s.newGame(count)
    x = 0
    while x != "FINISH":
        inputStr = int(input("Your Guess : "))
        x=s.guess(inputStr)
        print(x)
    guessCount = s.getGuessCount()
    print("SUCCESS in %d trials" %guessCount)
```

class-= Number

Methods=

- __init__(self)
- newGame(self, amount) : 새로운 게임을 시작
- guess(self, userGuess) : 맞추기 시도
- getGuessCount(self) : 현재까지 시도 회수 반환

Properties

- secret : 비밀 숫자들
- digits : 비밀 숫자의 자리수
- trials : 현재까지 시도 회수

서버의 내부 설계

2. 메서드의 구현

```
picknumber.py (~/example/Final-proj) - gedit
열기(O)  [icon]

import random

class Number:

    def __init__(self):
        self.secret = 0
        self.digit = 0
        self.trials = 0

    def newGame(self, amount):
        self.secret = random.randint(1, amount+1)
        self.digit = amount
        self.trials = 0

    def guess(self, userGuess):
        self.trials += 1
        if 0 < userGuess < self.secret:
            print "You have to type bigger number."
        elif userGuess > self.secret:
            print "You have to type smaller number."
        elif userGuess == 0:
            print "0 is out of range"
        elif userGuess == self.secret:
            print "FINISH"

    def getGuessCount(self):
        return self.trials
```

**생성자 매서드의 역할 : 객체의 인스턴스가 생길 때, 알려진 초기 상태에 있도록 한다.*

```
def __init__(self):
    self.secret = 0
    self.digit = 0
    self.trials = 0
```



Secret(비밀숫자),
digits(비밀 숫자의 자리
수), trials(현재까지 시도
회수)를 모두 0으로 설
정해 놓는다.

```
def newGame(self,
amount):
    self.secret =
random.randint(1,
amount+1)
    self.digit = amount
    self.trials = 0
```



비밀숫자의 범위를 1부
터 사용자가 입력한 값
(amount)까지로 설정하
고 범위안에서 랜덤으러
고르도록 한다. 비밀숫
자의 자리수도 amount
로 설정하고 시도횟수는
0으로 초기화한다.

서버의 내부 설계

2. 메서드의 구현

```
picknumber.py (~/.example/Final-proj) - gedit
열기(O)  [icon]

import random

class Number:

    def __init__(self):
        self.secret = 0
        self.digit = 0
        self.trials = 0

    def newGame(self, amount):
        self.secret = random.randint(1, amount+1)
        self.digit = amount
        self.trials = 0

    def guess(self, userGuess):
        self.trials += 1
        if 0 < userGuess < self.secret:
            print "You have to type bigger number."
        elif userGuess > self.secret:
            print "You have to type smaller number."
        elif userGuess == 0:
            print "0 is out of range"
        elif userGuess == self.secret:
            print "FINISH"

    def getGuessCount(self):
        return self.trials
```

```
def guess(self, userGuess):
    self.trials += 1
    if 0 < userGuess
    < self.secret:
        print "You have to
        type bigger number."
        elif userGuess >
        self.secret:
            print "You have to
            type smaller number."
            elif userGuess == 0:
                print "0 is out of
                range"
            elif userGuess ==
            self.secret:
                print "FINISH"
```



한 번 숫자를 입력해 문제를 풀 때마다 시도횟수를 1씩 더한다. userGuess가 0보다 크고 비밀숫자보다 작을 때 "You have to type bigger number."이라고 뜨고 비밀 숫자보다 클 경우 "You have to type smaller number."이라고 뜨게 한다. 0을 입력했을 경우 범위 안에 있지 않으므로 "0 is out of range"라고 뜨며 userGuess가 비밀숫자와 일치할 경우 "FINISH"라고 뜨게 한다.

서버의 내부 설계

2. 메서드의 구현

```
picknumber.py (~/example/Final-proj) - gedit
열기(O)  [icon]

import random

class Number:

    def __init__(self):
        self.secret = 0
        self.digit = 0
        self.trials = 0

    def newGame(self, amount):
        self.secret = random.randint(1, amount+1)
        self.digit = amount
        self.trials = 0

    def guess(self, userGuess):
        self.trials += 1
        if 0 < userGuess < self.secret:
            print "You have to type bigger number."
        elif userGuess > self.secret:
            print "You have to type smaller number."
        elif userGuess == 0:
            print "0 is out of range"
        elif userGuess == self.secret:
            print "FINISH"

    def getGuessCount(self):
        return self.trials
```

```
def getGuessCount(self):
    return self.trials
```



지금까지 맞추기를 시도
한 횟수를 반환한다.
(+ Getter 메서드와 같은
것)

서버의 내부 설계

3. 게임 구현의 테스트

```
picknumber.py (~/.final/final-proj) - gedit
열기(O)  [icon]

import random

class Number:

    def __init__(self):
        self.secret = 0
        self.digit = 0
        self.trials = 0

    def newGame(self, amount):
        self.secret = random.randint(1, amount)
        self.digit = amount
        self.trials = 0

    def guess(self, userGuess):
        self.trials += 1
        if 0 < userGuess < self.secret:
            return "You have to type bigger number."
        elif userGuess > self.secret:
            return "You have to type smaller number."
        elif userGuess == 0:
            return "0 is out of range"
        elif userGuess == self.secret:
            return "FINISH"

    def getGuessCount(self):
        return self.trials

if __name__ == '__main__':
    s = Number()
    count = int(input("Set Range : "))
    s.newGame(count)
    x = 0
    while x != "FINISH":
        inputStr = int(input("Your Guess : "))
        x=s.guess(inputStr)
        print(x)
    guessCount = s.getGuessCount()
    print("SUCCESS in %d trials" %guessCount)
```

`__name__ == '__main__'`



이 모듈이 최상위로 실행
되면 True
다른 모듈에 의해 import
되면 False

`s = Number()`



Number 클래스의 객체를
생성

`x = 0`



초기 `x = 0`

서버의 내부 설계

4. 게임 실행

```
kmucs@localhost:~$ python '/home/kmucs/final/final-proj/picknumber.py'  
Set Range : 5  
Your Guess : 3  
You have to type smaller number.  
Your Guess : 2  
FINISH  
input 5 digits!  
SUCCESS in 2 trials  
kmucs@localhost:~$
```

API의 구현

```
wsgi.py (~/example/Final-proj) - gedit
열기(O)  [icon]

picknumber.py  x    wsgi.py  x    Game.py

from cgi import parse_qs
import json
from game import new_game, guess

def application(environ, start_response):
    error = False

    if environ['REQUEST_METHOD'] != 'POST':
        response = {'code': 'error', 'msg': 'wrong HTTP method'}
        error = True

    if not error:
        try:
            path = environ['PATH_INFO'].split('/')
            if len(path) == 2:
                method = path[1]
            else:
                response = {'code': 'error', 'msg': 'wrong API path'}
                error = True
        except:
            response = {'code': 'error', 'msg': 'wrong API path'}
            error = True

    try:
        request_body_size = int(environ.get('CONTENT_LENGTH', '0'))
    except ValueError:
        request_body_size = 0

    request_body = environ['wsgi.input'].read(request_body_size)
    d = parse_qs(request_body)

    if not error:
        if method == 'new':
            response = new_game(d)
        elif method == 'guess':
            response = guess(d)
        else:
            response = {'code': 'error', 'msg': 'non-existent API method'}

    status = '200 OK'
    response_body = json.dumps(response)

    response_headers = [
        ('Content-Type', 'application/json'),
        ('Content-Length', str(len(response_body)))
    ]

    start_response(status, response_headers)

    return [response_body]
```

- ✓ If -> POST요청이 아니면 response=에러 메시지를 담은 사전
- ✓ 첫번째 if not error: 에서는 environ['PATH_INFO']로부터 요청된 API를 파악하고 request body를 읽어 들여 query string으로 해석한다.
- ✓ 두번째 if not error: 에서는 요청된 API에 따라 적당한 함수를 호출하고 json형태를 가지는 HTTP response를 출력한다.

API의 구현

```
def application(environ, start_response):  
    error = False  
  
    if environ['REQUEST_METHOD'] != 'POST':  
        response = {'code': 'error', 'msg': 'wrong HTTP method'}  
        error = True  
  
    if not error:  
        try:  
            path = environ['PATH_INFO'].split('/')  
            if len(path) == 2:  
                method = path[1]  
            else:  
                response = {'code': 'error', 'msg': 'wrong API path'}  
                error = True  
        except:  
            response = {'code': 'error', 'msg': 'wrong API path'}  
            error = True  
  
    try:  
        request_body_size = int(environ.get('CONTENT_LENGTH', '0'))  
    except ValueError:  
        request_body_size = 0  
  
    request_body = environ['wsgi.input'].read(request_body_size)  
    d = parse_qs(request_body)
```

POST 요청인지 판단하고 아니면 에러 응답을 준비하고 다음 단계들은 실행하지 않도록 한다.

Environ에 주어진 환경변수로부터 request path를 검사해 어떤 API가 호출되었는지 판단한다.

- ✓ 입력받은 값이 없을까봐 except ValueError코드를 넣었다.
- ✓ Request_body = environ['wsgi.input'].read(request_body_size)
d=parse_qs(request_body) 코드는 d는 POST 값을 받아온다.

API의 구현

```
if not error:
    if method == 'new':
        response = new_game(d)
    elif method == 'guess':
        response = guess(d)
    else:
        response = {'code': 'error', 'msg': 'non-existent API method'}

status = '200 OK'
response_body = json.dumps(response)

response_headers = [
    ('Content-Type', 'application/json'),
    ('Content-Length', str(len(response_body)))
]

start_response(status, response_headers)

return [response_body]
```

두 함수 new_game(d)와 guess(d)는 게임 드라이버 모듈이 제공

Response는 dictionary타입의 객체이다.
내용에는 게임실행에 대한 결과값이 들어있다.

게임 드라이버의 구현

```
Game.py (~/example/Final-proj) - gedit
열기(O) [icon]

from picknumber import Number

skm = Number()

def guess(d):
    try:
        guess = int(d.get('guess', [''])[0])
    except:
        return {'code': 'error', 'msg': 'Wrong Guess'}

    answer = skm.guess(guess)
    trials = skm.getGuessCount()

    return {'code': 'success', 'guess': guess, 'answer': answer, 'trials': trials}

def new_game(d):
    try:
        count = int(d.get('count', [''])[0])
    except:
        return {'code': 'error', 'msg': 'Count Not Given'}

    skm.newGame(count)
    return {'code': 'success'}
```

게임에 이용되는 메인 클래스(Number)를 import 한다.

Number 클래스의 객체 하나를 skm으로 초기화한다.

Guess()함수와 new_game()함수를 구현한다.

게임 드라이버의 구현 새로운 게임(new_game)

```
def new_game(d):  
    try:  
        count = int(d.get('count', [''])[0])  
    except:  
        return {'code': 'error', 'msg': 'Count Not Given'}  
  
    skm.newGame(count)  
    return {'code': 'success'}
```

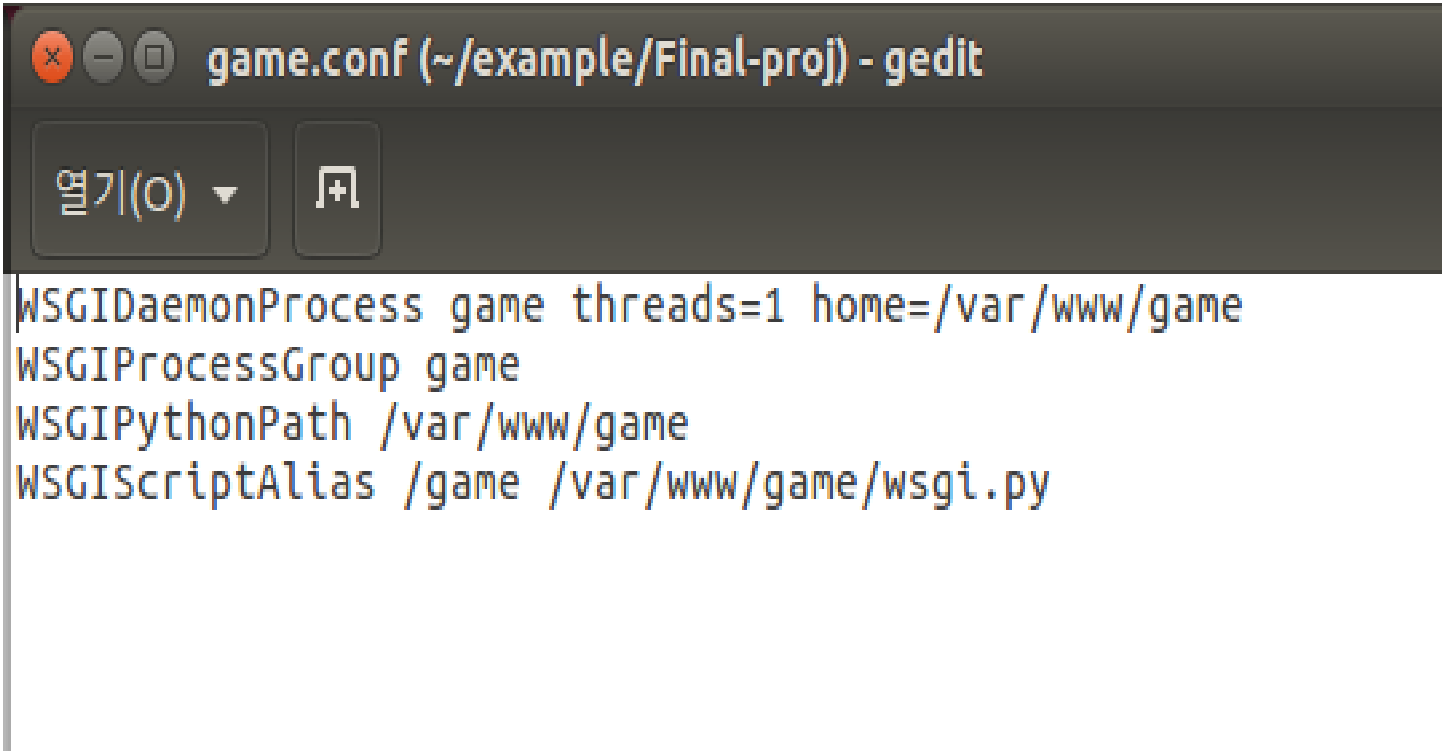
- ✓ 입력한 값이 오류가 발생할까봐 try 코드를 작성했다.
- ✓ 새로운 게임을 시작하도록 skm.newGame(count)를 작성했다.
- ✓ 성공하는 경우에 대한 응답을 만들어 반환하기 위해 return을 사용했다.

게임 드라이버의 구현 맞추기 시도(guess)

```
def guess(d):  
    try:  
        guess = int(d.get('guess', [''])[0])  
    except:  
        return {'code': 'error', 'msg': 'Wrong Guess'}  
  
    answer = skm.guess(guess)  
    trials = skm.getGuessCount()  
  
    return {'code': 'success', 'guess': guess, 'answer': answer, 'trials': trials}
```

✓ 입력한 값이 오류가 발생할까봐 try 코드를 작성했다.

Apache 서버의 WSGI 설정

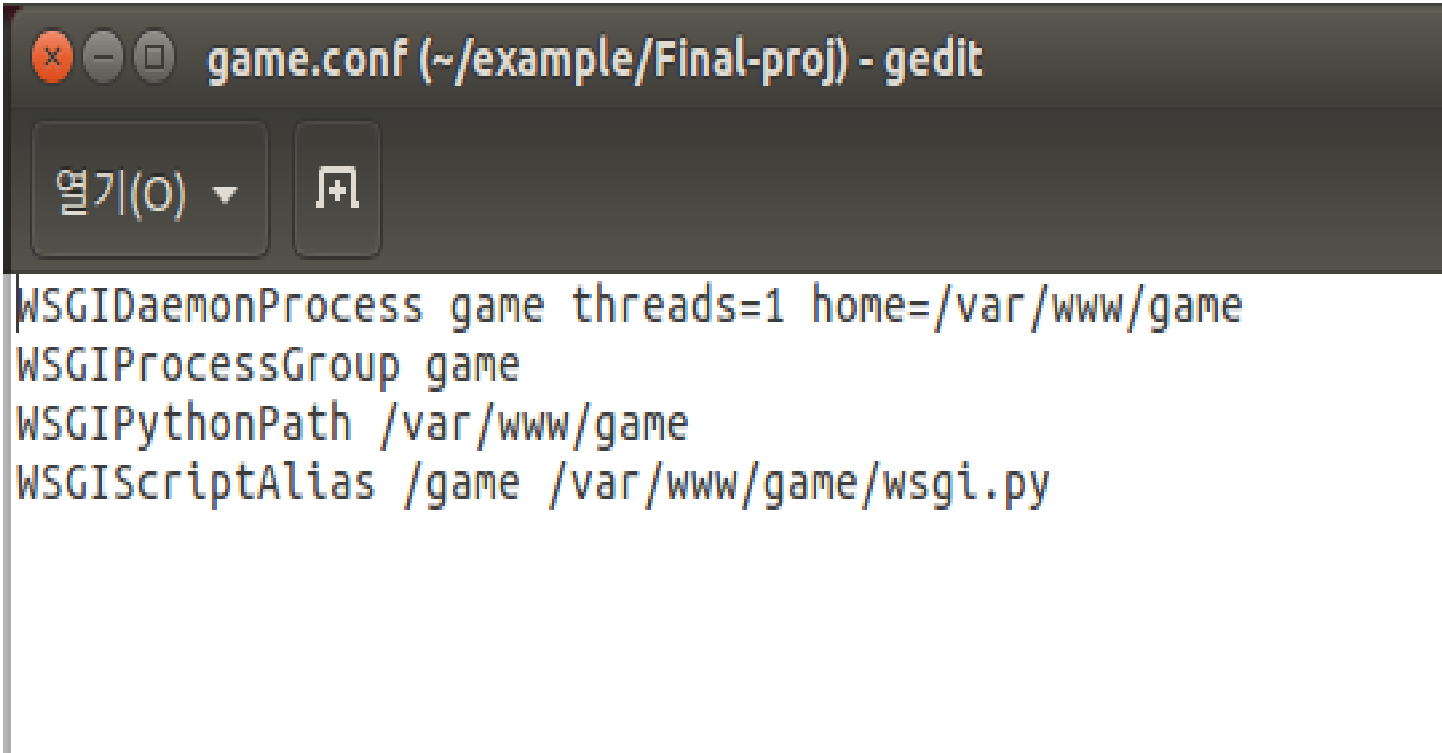


The screenshot shows a gedit editor window titled "game.conf (~/.example/Final-proj) - gedit". The window has a menu bar with "열기(O)" and a search icon. The main text area contains the following configuration lines:

```
WSGIDaemonProcess game threads=1 home=/var/www/game
WSGIProcessGroup game
WSGIPythonPath /var/www/game
WSGIScriptAlias /game /var/www/game/wsgi.py
```

- ✓ 별도의 daemon 프로세스가 처리하도록 하고 thread는 한 개만 한다.
- ✓ Game.py와 strikeball.py내의 객체들을 import 할 수 있도록 하였다.

서버의 테스트

A screenshot of a gedit text editor window. The title bar shows the file path as 'game.conf (~/.example/Final-proj) - gedit'. Below the title bar is a menu bar with '열기(O)' (Open) and a search icon. The main text area contains the following configuration lines:

```
WSGIDaemonProcess game threads=1 home=/var/www/game
WSGIProcessGroup game
WSGIPythonPath /var/www/game
WSGIScriptAlias /game /var/www/game/wsgi.py
```

- ✓ 별도의 daemon 프로세스가 처리하도록 하고 thread는 한 개만 한다.
- ✓ Game.py와 strikeball.py내의 객체들을 import 할 수 있도록 하였다.

터미널에서 curl 이용해 기본 동작 테스트

```
localhost: ~
```

```
kmucs@localhost:~$ curl -d"count=3" http://10.30.100.8/game/new
```

```
{"code": "success"}kmucs@localhost:~$
```

```
kmucs@localhost:~$ curl -d"guess=2" http://10.30.100.8/game/guess
```

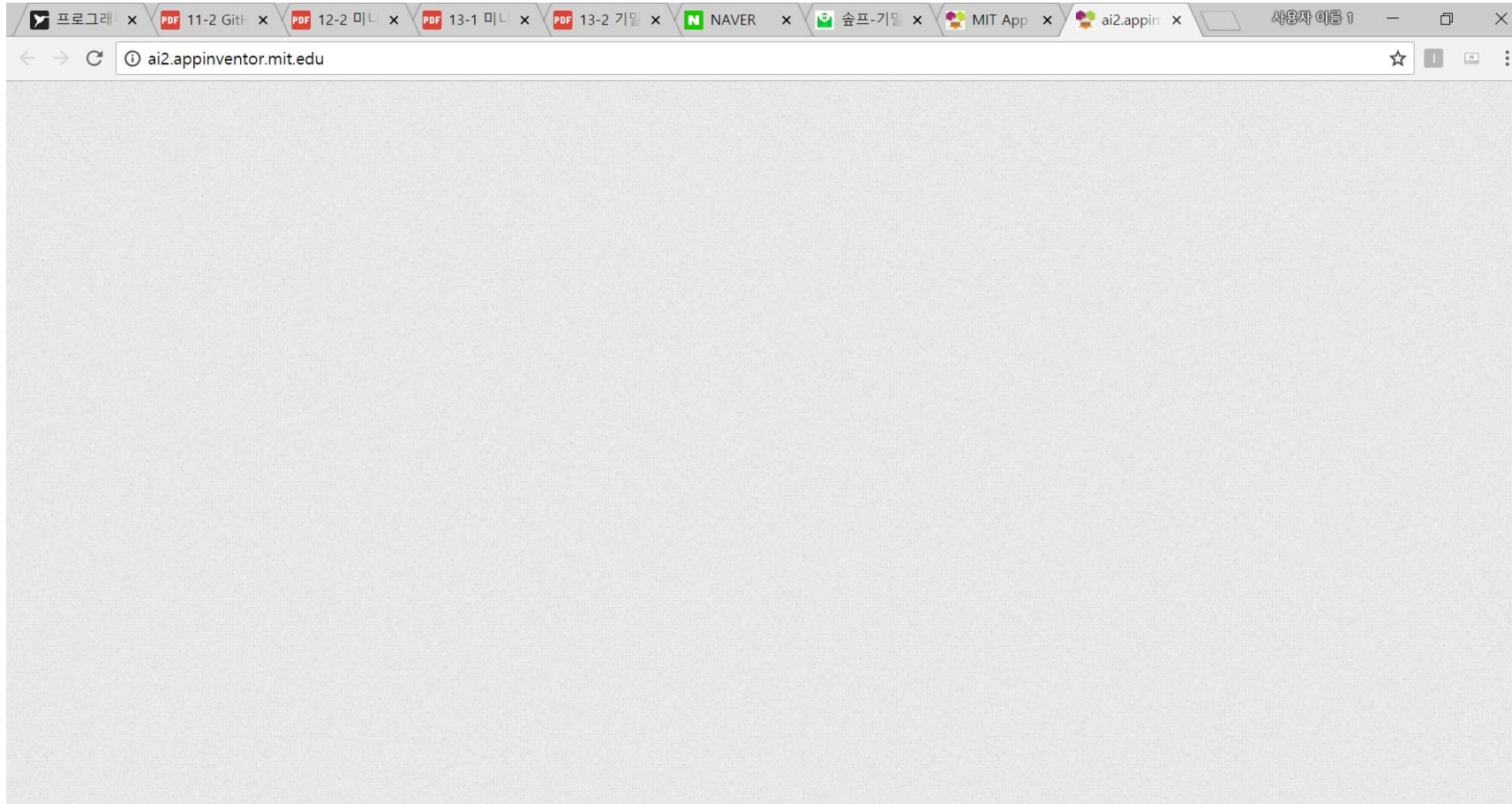
```
{"answer": "Please type bigger number.", "trials": 1, "guess": 2, "code": "success"}kmucs@localhost:~$ curl -d"guess=3" http://10.30.100.8/game/guess
```

```
{"answer": "Please type bigger number.", "trials": 2, "guess": 3, "code": "success"}kmucs@localhost:~$ curl -d"guess=4" http://10.30.100.8/game/guess
```

```
{"answer": "END", "trials": 3, "guess": 4, "code": "success"}kmucs@localhost:~$ curl -d"guess=4" ht
```

```
kmucs@localhost:~$
```


게임 클라이언트 개발-App inventor



코드를 다 짰 후 앱인벤터를 하려고 했으나 계속 앱인벤터가 열리지 않아 블록 코딩을 하지 못했습니다.

Github에 제출하기

```
kmucs@localhost: ~/final
kmucs@localhost:~$ cd final
kmucs@localhost:~/final$ git add final-proj
kmucs@localhost:~/final$ git commit -m "My second commit"
[master 3ef46d5] My second commit
1 file changed, 9 insertions(+), 12 deletions(-)
kmucs@localhost:~/final$ git push origin master
Username for 'https://github.com': skm0626@kookmin.ac.kr
Password for 'https://skm0626@kookmin.ac.kr@github.com':
오브젝트 개수 세는 중: 4, 완료.
Delta compression using up to 4 threads.
오브젝트 압축하는 중: 100% (3/3), 완료.
오브젝트 쓰는 중: 100% (4/4), 424 bytes | 0 bytes/s, 완료.
Total 4 (delta 2), reused 0 (delta 0)
remote: Resolving deltas: 100% (2/2), completed with 2 local objects.
To https://github.com/S3-20181630/20181630-final-project.git
   f1f6090..3ef46d5  master -> master
kmucs@localhost:~/final$
```

- ✓ 별도의 디렉토리 만들어 파일배치
- ✓ Git add
- ✓ Git commit
- ✓ Git push

Github에서 확인하기

S3-20181630 / 20181630-final-project

Watch

0

Star

0

Fork

0

Code

Issues0

Pull requests0

Projects0

Wiki

Insights

Settings

Branch: master

20181630-final-project / final-proj /

Create new file

Upload files

Find file

History

S3-20181630 My second commit

Latest commit 3ef46d5 9 hours from now

..

Game.py

My first commit

just now

game.conf

My first commit

just now

picknumber.py

My second commit

just now

wsgi.py

My first commit

just now