CGI와 웹서버

seanlion · 2020년 8월 2일

서버

파이썬에 대해 다시 본격적으로 공부하기로 마음먹으면서 바로 생활코딩의 파이썬 코스를 수강하기 시작했다.

블로그의 첫 시리즈에서는 해당 코스를 수강하면서 가장 많이 공부한 개념인 CGI에 대해 다루고자 한다.

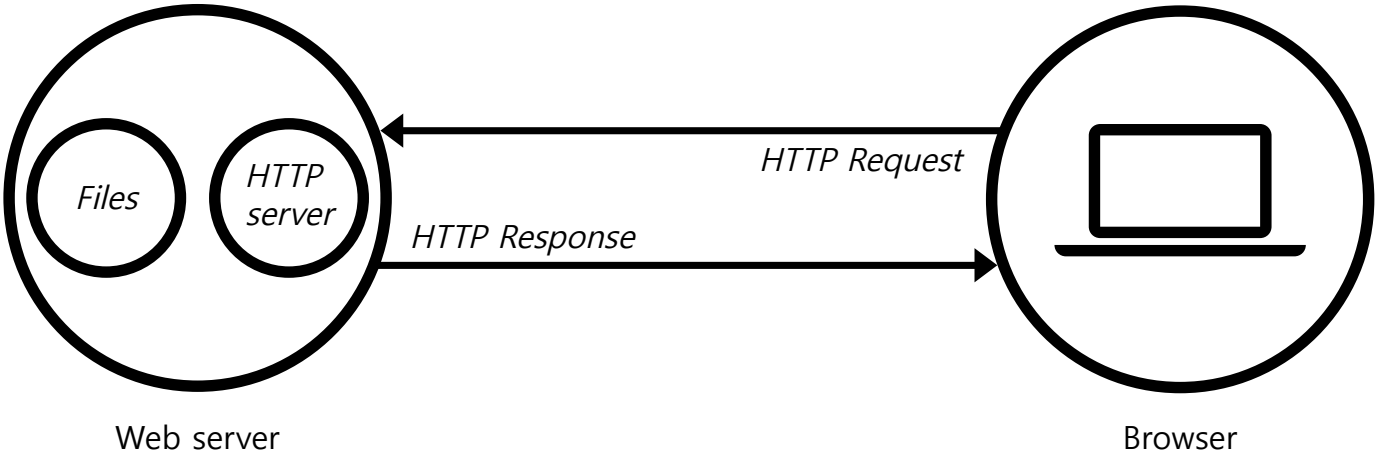
CGI와 웹서버

- CGI란 공통 게이트웨이 인터페이스(Common Gateway Interface)의 약어로, 웹서버와 외부 프로그램 사이에서 정보를 주고받는 방법이나 규약들을 말한다.

CGI를 알려면 공부할 개념들이 몇가지 있는데, 일단 웹서버에 대해서만 간략하게 알아보자.

웹 서버의 기본적인 개념은 mozllia.org에서 잘 설명해놓아서 그 내용을 참고했다.

> 소프트웨어 측면에서, **web server는 기본적으로 웹 사용자가 어떻게 호스트 파일들에 접근하는지를 관리합니다.** 여기서 web server는 HTTP서버로 국한합니다.
> HTTP 서버는 URL(Web address)과 HTTP(라우저가 웹 페이지를 보여주기 위해 사용하는 프로토콜)의 소프트웨어 일부입니다.
> 브라우저가 웹 서버에 있는 파일을 필요로 할때, 브라우저는 HTTP를 통해 파일을 요청합니다.
> 요청이 올바른 웹 서버(하드웨어)에 도달하였을 때, HTTP 서버(software)는 요청된 문서를 HTTP를 이용해 보내줍니다.



웹 서버는 정적 웹서버와 동적 웹서버로 나뉜다.

정적 웹 서버 :

- HTTP 서버가 있는 컴퓨터로 구성.
- 서버에 존재하는 이미 저장된 파일(HTML, 스크립트 등)을 브라우저에게 전송.
- 서버에 저장된 데이터가 변경되지 않는 한 고정된 웹 페이지를 보게 됨.

동적 웹 서버

- 정적 웹 서버와 어플리케이션 서버(AS)로 구성.
- AS는 웹 서버에서 처리하지 못하는 동적 데이터에 대응하기 위해 만들어진 어플리케이션 서버(데이터베이스 조회, 로직 처리 등)
- 어플리케이션 서버는 프로그램에게 응답을 전달 받아 웹 서버에 전달하게 됨.
- WAS는 웹 서버 + 어플리케이션 서버를 포함하는 개념.(동적 서버 콘텐츠를 수행하는 것으로 정적 콘텐츠를 전달하는 일반적인 웹 서버와는 구별 하는 것 같다.)

- 어플리케이션 서버는 하나의 프로토콜로서 CGI와 유사한 기능을 수행한다고 봐야할 것 같다.

프로세스 :
WAS가 웹 서버로부터 처리 요청을 받으면, 프로그램의 실행결과를 웹 서버에 전달 → 웹 서버는 해당 결과를 웹 클라이언트에 전송

클라이언트 ↔ WAS 프로세스를 잘 정리한 글을 참고하면 좋을 것 같다.

대표적인 웹서버로는 Apache HTTP Server, nginx가 있고, AS에는 Apache Tomcat이 있다.

정리

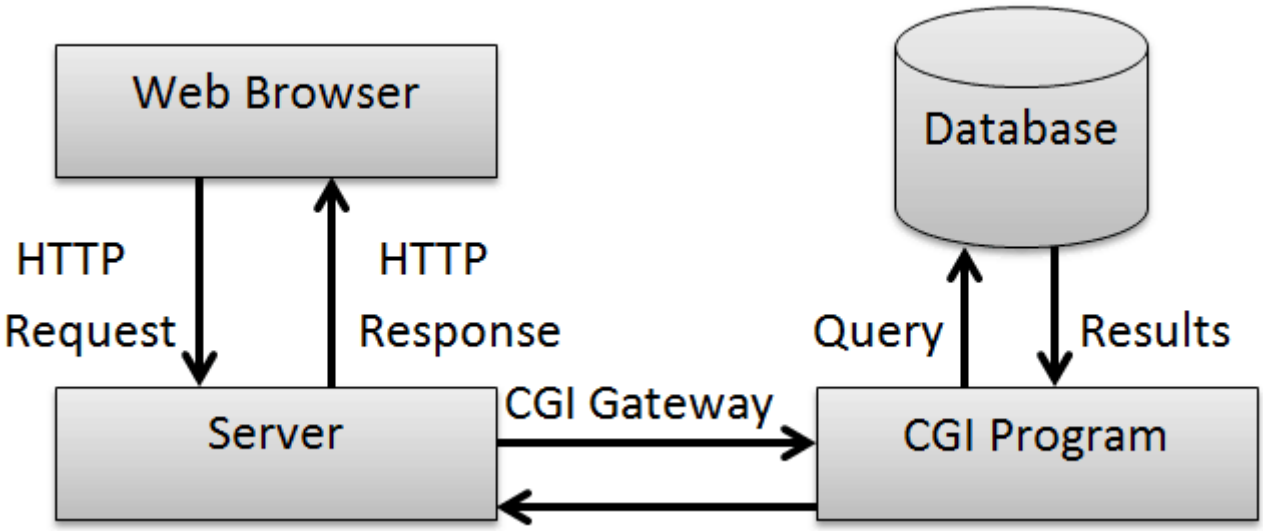
그렇다면 이제 CGI에 대해 정리해보자.

- CGI는 (위에서 정의한대로) 정보를 주고받는 방법이나 규약이다.
  웹 서버도 종류가 여러가지일 것이고, 프로그램 또한 엄청나게 많은 프로그램이 존재하기 때문에 서로 입출력을 주고 받을 표준이 필요하다.
- 이 표준에 맞추어 만들어진 것이 CGI 스크립트이다. CGI 스크립트는 어떤 프로그래밍 언어로도 만들 수 있다.
  (C, Perl, Python 등 으로 구현 가능.)

> CGI에는 '인터페이스'라는 개념이 들어가기 때문에 인터페이스에 대해서도 간단하게 찾아보았다.
> 인터페이스 = **상호 간의 소통을 위해 만들어진 물리적 매개체나 프로토콜**

> 그럼 프로토콜이란? : 컴퓨터나 원거리 통신 장비 사이에서 메시지를 주고 받는 양식과 규칙의 체계

아래는 CGI를 잘 표현해놓은 이미지이다.



> 여기서 HTTP(Protocol)이란 인터넷 통신규약 중 하나로서 브라우저에서 HTML 파일을 전송해주는 규약을 말합니다.

이렇게 보면, CGI는 위에서 설명한 AS(어플리케이션 서버) 개념과 유사하다고 볼 수 있다. 나는 기본적으로는 웹서버가 클라이언트에게 응답을 보내기 위해 사용자가 만든 애플리케이션과 통신 해야 하는데, 그것을 도와주는 인터페이스라고 이해했다.

실제 CGI를 통해 동적인 콘텐츠를 전달하는 걸 간단하게 살펴보자.

- 서버의 cgi-bin이라는 폴더를 만들어놓고, 그 내부의 스크립트 파일을 만들어놓는다.
- 웹서버가 CGI를 통해 cgi bin에 접속해서 그 내부의 파일을 실행시키고, 그 결과를 클라이언트에 보낸다.

좀 더 깊게 보려면, 아래 링크에 있는 이미지를 간략하게 보면 도움이 될 것 같다.

> https://parkansky.com/tutorials/bdlogcgi.htm

CGI의 특징

CGI의 특징을 이해하는데 이 글의 도움을 많이 받았다.

- CGI는 가장 오래된 인터페이스이고, 거의 모든 웹서버를 지원 가능.
- CGI를 구동하는 방법이 한가지는 아니지만, 대표적인 방법이 Apache HTTPd.

- 웹서버와 통신하기 위해 CGI를 사용하는 프로그램은 매 리퀘스트마다 서버를 재시작해야 함.
- 모든 리퀘스트는 파이썬 인터프리터를 새롭게 구동하기 때문에 CGI는 부하가 적은 상황에서만 쓸모 있는듯.
- CGI의 장점은 간편하다는 것. CGI를 사용하는 프로그램을 작성할 때 세줄 정도만 추가하면 됨.(그러나, 이러한 간편성은 댓가를 치를 수 있음.)
- 단점도 명확한 편. 파이썬 스크립트가 처음부터 실행 가능하지 않음. CGI 스크립트가 실행 가능하지 않은 상태이면, 웹서버는 유저에게 이 스크립트를 실행시키고 결과를 전송하는게 아니라 다운로드 하게 함. 그래서 파일 생성시 매번 파일의 실행 권한을 얻을 수 있는 `chmod a+x your_script.py` 를 실행 시켜주어야함.
- 스크립트 첫 라인은 무조건 Unix-type이여야 한다고 함.(shebang)
  나는 Mac OS 환경이어서 많이 귀찮진 않았지만 윈도우 환경은 Unix line endings으로 변환이 필요하다고 함. (다행히 대부분의 에디터가 이 변환을 지원한다고 함.)
- 또한 shebang 안의 path가 항상 정확해야함. shebang은 간단히 말하면 Python path를 찾는 코드인데, 정의해놓은 path에 Python이 설치가 안 되어있으면 코드가 동작 하지 않음. 그래서 항상 `#!/usr/bin/env python` 등의 경로를 숙지하고 있어야 함.
- 웹서버가 파일을 읽을 수 있어야 하기 때문에 파일을 항상 읽기 가능한 권한으로 만들어야 함.
- 스크립트 파일이 Byte Order Mark를 포함하지 않아야 함. (BOM은 UTF-16 관련 개념인 것 같은데, 좀 더 공부해야봐야겠다.)

> CGI 이후에 WSGI라는 개념이 나왔는데, 현재는 WSGI를 통해 CGI를 구동하는 프로그램을 작성하는게 가능함. 더 나은 옵션이 없을 때 WSGI를 통해 CGI를 구동하면 된다.

마치며

CGI를 공부하다보니 웹서버에 대해서도 많이 알게 되었다. CGI 이후에도 발전을 위해 mod_python, FastCGI, SCGI, WSGI 라는 개념도 나왔는데 후속 글에서 다루어보고자 한다.

참고 글 :

1. https://www.nginx.com/resources/glossary/application-server-vs-web-server/
2. https://sfeg.tistory.com/196
3. https://stackoverflow.com/questions/2089271/what-is-common-gateway-interface-cgi
4. https://information-science.tistory.com/2

승톨

소프트웨어 엔지니어링을 연마하고자 합니다.

다음 포스트
CGI의 발전에 대해 알아보자.

이전 포스트
cgi.FieldStorage와 인스턴스

3개의 댓글

댓글을 작성하세요

댓글 작성

jewelrykim
2021년 11월 17일

개발 도서 스터디 자료 만들던 중에 우연히 들어오게 되었습니다 ^&^ 좋은 자료 감사합니다 ㅎㅎ

2개의 답글