02 URL과 리소스

CodeDiary18(codediary18@gmail.com)

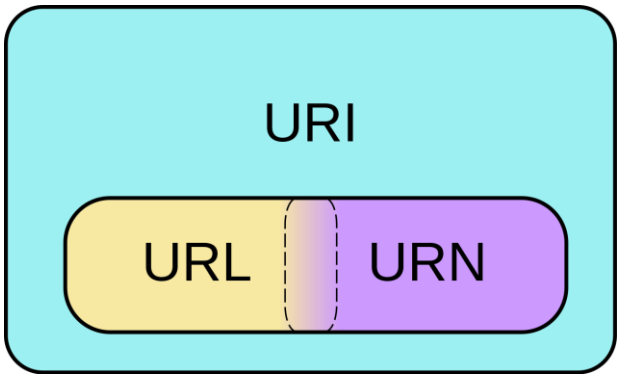
2022.02.17

목차

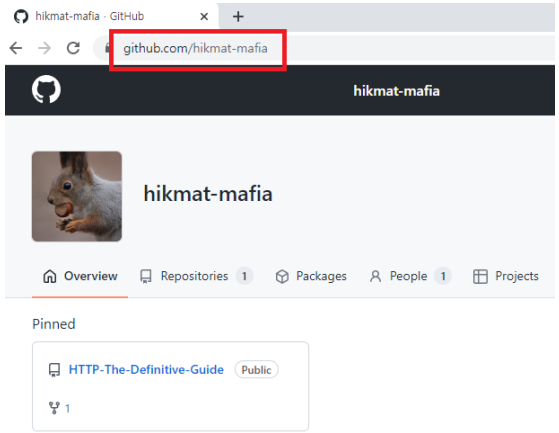
2.1 인터넷의 리소스 탐색하기

✓ **URL**(Uniform Resource Locator)이란?

- 인터넷의 리소스를 가리키는 표준 이름
- 웹 상의 리소스가 어디에 있고 어떻게 접근할 수 있는지 알려주는 역할
- URI의 부분집합

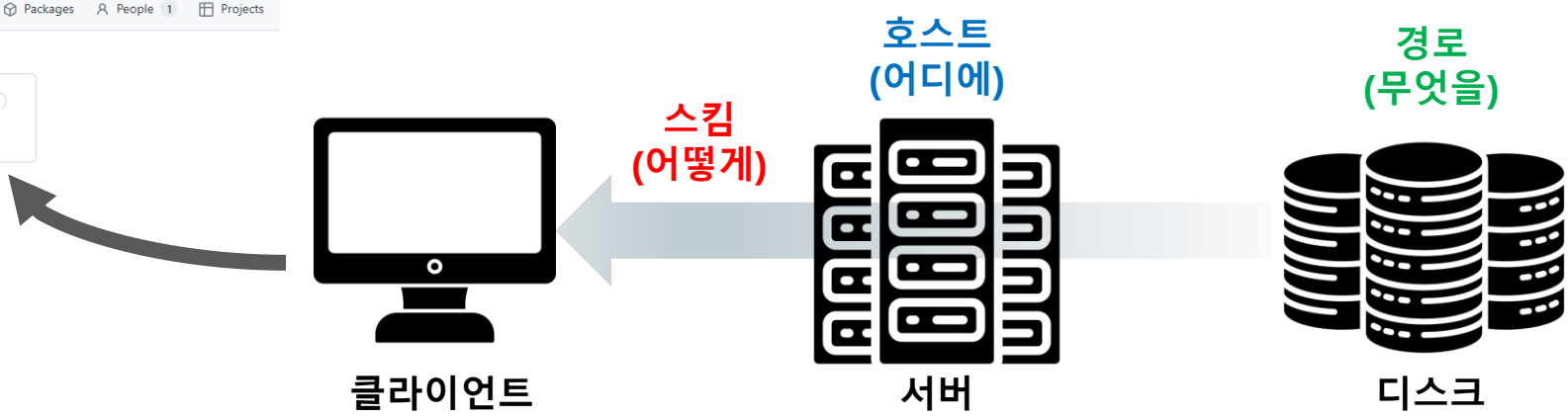


2.1 인터넷의 리소스 탐색하기



스킴(어떻게)://호스트(어디에)/경로(무엇을)

https://github.com/hikmat-mafia.html



2.1 인터넷의 리소스 탐색하기

- ✓ HTTP 프로토콜이 아닌 다른 가용한 프로토콜도 사용가능
  - mailto:abc@google.com : 메일
  - ftp://ftp.books.com/pub/test.xls : ftp 서버에 올라가 있는 파일
  - rtsp://www.google.com/image/5 : 스트리밍

2.2 URL 문법

URL의 기본 형태

<스킴>://<사용자이름>:<비밀번호>@**<호스트>**:<포트>/**<경로>**;<파라미터>?<질의>#<프래그먼트>

- 세부적인 형태는 스킴에 따라 달라짐
- 모든 컴포넌트를 가지는 url은 거의 없음
- 주로 스킴, 호스트, 경로가 중요함

컴포넌트	설명	기본값
스킴	서버에 접근하는 프로토콜 지시	
사용자 이름	몇몇 스킴은 리소스에 접근하기 위해 사용자 이름이 필요	anonymous
비밀번호	사용자의 비밀번호	<이메일주소>
호스트	리소스를 호스팅하는 서버의 호스트 명이나 IP 주소	
포트	리소스를 호스팅하는 서버가 열어 놓은 포트 번호. 많은 스킴이 기본 포트를 가지고 있음 (HTTP의 기본 포트는 80)	스킴에 따라 상이
경로	서버 내 리소스가 어디에 있는지 가리킴	
파라미터	이름/값을 쌍으로 가짐	
질의	스킴에서 애플리케이션에 파라미터를 전달하는데 쓰임. URL의 끝에 '?'로 구분	
프래그먼트	리소스의 조각이나 일부분을 가리킴. 클라이언트에서만 사용. URL의 끝에 '#'으로 구분	

2.2 URL 문법

<스킴>://<사용자이름>:<비밀번호>@<호스트>:<포트>/<경로>;<파라미터>?<질의>#<프래그먼트>

- ✓ 스킴 : **사용할 프로토콜**
  - 주어진 리소스에 어떻게 접근하는지 알려주는 중요한 정보
    - 어떤 프로토콜을 사용하여 리소스를 요청해야 하는지
  - 스킴 컴포넌트는 알파벳으로 시작해야 하고 URL의 나머지 부분들과 ':'로 구분
  - 스킴명은 대소문자를 구분하지 X
    - HTTP://github.com == http://github.com

2.2 URL 문법

<스킴>://<사용자이름>:<비밀번호>@**< 호스트>:<포트>**/<경로>;<파라미터>?<질의>#<프래그먼트>

✓ 호스트와 포트

- 리소스를 호스팅하고 있는 장비와 장비 내에서 리소스에 접근할 수 있는 서버가 어디에 있는지 알려주는 역할
- **호스트 컴포넌트**는 접근하려고 하는 리소스를 가지고 있는 인터넷 상의 호스트 장비를 가리킴
  - **호스트 명이나 IP 주소**로 제공

https://github.com/hikmat-mafia.html
https://15.164.81.167:443/hikmat-mafia.html

- **포트 컴포넌트**는 서버가 열어 놓은 **네트워크 포트**를 가리킴
  - 내부적으로 TCP 프로토콜을 사용하는 HTTP는 기본 포트로 80을 사용

2.2 URL 문법

<스킴>://**<사용자이름>:<비밀번호>**@< 호스트>:<포트>/<경로>;<파라미터>?<질의>#<프래그먼트>

- ✓ 사용자 이름과 비밀번호
  - 데이터 접근을 위해 사용자 이름과 비밀번호를 요구하는 경우 사용됨
  - FTP 서버가 많이 사용함
    - 사용자 이름과 비밀번호를 기술하지 않고 FTP URL에 접근하면, 기본 사용자 이름 값으로 'anonymous' 비밀번호는 브라우저마다 가지고 있는 기본값을 사용

```
ftp://ftp.prep.ai.mit.edu/pub/gnu
ftp://anonymous@ftp.prep.ai.mit.edu/pub/gnu
ftp://anonymous:my_passwd@ftp.prep.ai.mit.edu/pub/gnu
```

2.2 URL 문법

<스킴>://**<사용자이름>:<비밀번호>**@< 호스트>:<포트>/**<경로>**;<파라미터>?<질의>#<프래그먼트>

- ✓ 경로
  - 리소스가 서버의 어디에 있는지 알려줌
    - 서버가 리소스의 위치를 찾는데 사용하는 정보
  - 유닉스 파일 시스템의 파일 경로와 유사
  - HTTP URL에서 경로 컴포넌트는 '/' 문자를 기준으로 경로 조각으로 나뉨
    - 각 경로 조각은 자체만의 파라미터 컴포넌트를 가질 수 있음

2.2 URL 문법

<스킴>://**<사용자이름>:<비밀번호>**@< 호스트>:<포트>/<경로>;**<파라미터>**?<질의>#<프래그먼트>

✓ 파라미터

- 애플리케이션이 **서버에 정확한 요청**을 하기 위해 필요한 입력 파라미터를 받는데 사용
- 이름/값 쌍의 리스트로 URL 나머지 부분들로부터 ';' 문자로 구분하여 URL에 기술
  - http://www.google.com/image;**name=cute**
    - 이름은 'name'이고 값은 'cute'인 name=cute라는 하나의 파라미터 전달
  - http://www.google.com/hammers;**sale=false**/index.html;**graphics=true**
    - hammers 경로 조각은 값이 false인 sale 파라미터를 가짐
    - index.html 경로 조각은 값이 true인 graphics라는 파라미터를 가짐

2.2 URL 문법

<스킴>://**<사용자이름>:<비밀번호>**@< 호스트>:<포트>/<경로>;<파라미터>?**<질의>**#<프래그먼트>

- ✓ 질의 문자열
  - 데이터베이스 같은 서비스들은 **요청받을 리소스 형식의 범위를 좁히기** 위해서 질문이나 질의를 받을 수 있음
  - 게이트웨이를 가리키는 URL의 경로 컴포넌트와 함께 전달
    - 게이트웨이는 다른 애플리케이션에 접근하려고 할 때 거치는 통로라고 할 수 있음
  - 많은 게이트웨이가 '&'로 나뉘는 'key=value' 형식의 질의 문자열을 원함
    - https://www.google.com/search?**q=질의&source=lnms**

2.2 URL 문법

<스킴>://**<사용자이름>:<비밀번호>**@< 호스트>:<포트>/<경로>;<파라미터>?<질의>**#<프래그먼트>**

- ✓ 프래그먼트
  - 리소스 내의 특정 부분(조각)을 가리킴
  - 일반적으로 HTTP 서버는 객체 일부가 아닌 전체만 다룸
    - 따라서, 클라이언트는 서버에 프래그먼트를 전달 X
    - 브라우저가 서버로부터 전체 리소스를 내려받은 후, 프래그먼트를 사용하여 보고자 하는 리소스의 일부를 보여줌
    - https://github.com/hikmat-mafia/HTTP-The-Definitive-Guide**#v-콘텐츠-발행-및-배포**

2.3 단축 URL

- ✓ URL은 상대 URL과 절대 URL로 나뉨
- ✓ 절대 URL: 리소스에 접근하는데 필요한 모든 정보를 가짐
- ✓ 상대 URL: 모든 정보를 담고 있지는 않은 URL을 짧게 표현하는 방식
  - 필요한 모든 정보를 얻기 위해서는 기저 (base) URL 사용
  - 스킴, 호스트 및 다른 컴포넌트를 모두 입력하지 않아도 됨

```
http://www.joes-hardware.com # 기저 URL
./hammers.html # 상대 URL
=> http://www.joes-hardware.com/hammers.html # 새로운 절대 URL
```

같은 서비스 내에서 페이지만 이동하고 싶다면 상대 URL 사용하면 좋음

2.3 단축 URL

✓ URL 확장

- 호스트명 확장

  주소 입력란에 ‘google’입력시, 브라우저가 호스트 명에 자동으로 ‘www.‘과 ‘.com’을 붙여서 ‘www.google.com’을 생성

- 히스토리 확장

  과거에 사용자가 방문했던 URL의 기록을 저장

2.4 안전하지 않은 문자

- ✓ 안전한 전송이란?
  정보가 유실될 위험 없이 URL을 전송할 수 있다는 것을 의미

- ✓ URL은 상대적으로 작고 일반적으로 안전한 알파벳 문자만 포함하도록 허락

- ✓ 알파벳 이외의 문자를 포함할 수 있도록 이스케이프라는 기능을 추가하여,
  안전하지 않은 문자를 안전한 문자로 인코딩할 수 있게 함

2.4 안전하지 않은 문자

✓ URL 문자 집합

- 역사적으로 많은 애플리케이션이 US-ASCII 문자 집합 사용; 만들어진 지 오래되어 표현 가능한 문자의 수가 적음
- URL이 특정 이진 데이터를 포함해야 하는 경우 존재
- 따라서 이스케이프 문자열을 쓸 수 있도록 설계
- US-ASCII에서 사용이 금지된 문자들로, 특정 문자나 데이터를 인코딩할 수 있게 함으로써 이동성과 완성도를 높임

2.4 안전하지 않은 문자

✓ 인코딩 체계

- URL의 안전하지 않은 문자를 인코딩하기 위해 퍼센티지 기호(%)로 시작해 ASCII 코드로 표현되는 두 개의 16진수 숫자로 이루어진 이스케이프 문자로 변환

문자	ASCII 코드	URL의 예
~	126 (0x7E)	http://github.hikmat-mafia.io/%7Eha
빈 문자	32 (0x20)	http://github.hikmat-mafia.io/more%20about.html
%	37 (0x25)	http://github.hikmat-mafia.io/100%25happy.html

2.4 안전하지 않은 문자

✓ 문자 제한

- URL 내에서 특별한 의미로 예약된 문자 존재
- 예약 문자를 다른 목적으로 사용하려면 그 전에 반드시 인코딩 필요

문자	선점 및 제한
%	이스케이프 토큰
/	경로 세그먼트 분리
.	경로 컴포넌트에서 선점
..	
#	프래그먼트의 구획 문자로 선점
?	질의 문자열 구획 문자
;	파라미터 구획 문자
:	스킴, 이름/비밀번호, 호스트/포트 구획 문자
$, +	선점
@&=	특정 스킴에서 의미가 있기 때문에 선점
{} | \ . ~ [] `	게이트웨이 등 여러 전송 에이전트에서 불안전하게 다루기 때문에 제한됨

2.4 안전하지 않은 문자

✓ 좀 더 알아보기

- 입력받은 URL에서 어떤 문자를 인코딩해야 하는지 결정하는 데는 브라우저와 같이 사용자로부터 최초로 URL을 입력받는 애플리케이션에서 하는 것이 가장 적절
- URL을 해석하는 애플리케이션은 그것을 처리하기 전에 URL을 디코드 해야함

2.5 스킴의 바다

- ✓ 웹에서 쓰이는 일반 스킴들의 포맷
  - http
  - https
  - mailto
  - ftp
  - rtsp, rtspu
  - file
  - news
  - telnet

2.5 스킴의 바다

- ✓ http(Hypertext Transfer Protocol)
  - 사용자이름 / 비번 외에 모든 컴포넌트를 사용
  - 포트값이 생략되어 있으면 기본값은 80
  - 기본형식
    http://<호스트>:<포트>/<경로>?<질의>#<프래그먼트>

2.5 스킴의 바다

- ✓ https
  - http 스킴과 거의 같음
  - HTTP의 커넥션의 양 끝단에서 암호화하기 위해 넷스케이프에서 개발한 소켓 계층(Secure Sockets Layer, SSL) 사용
  - 기본 포트값은 443
  - 기본형식
    https://<호스트>:<포트>/<경로>?<질의>#<프래그먼트>

2.5 스킴의 바다

- ✓ mailto
  - 이메일 주소를 가리킴
  - 다른 스킴과는 다르게 동작하므로, 표준 URL과 다른 포맷을 가짐
  - 문법은 RFC 822에 기술되어 있음
  - 기본형식
    mailto: <RFC-822-addr-spec>
  - 예
    mailto:test@gmail.com

2.5 스킴의 바다

- ✓ ftp(File Transfer Protocol) : 파일 전송 프로토콜
  - FTP서버의 파일을 내려 받거나 올리고, 콘텐츠 목록을 가져올 때 사용
  - FTP는 웹과 URL이 등장하기 전부터 있었음
  - 기본형식
    ftp://<사용자이름>:<비밀번호>@<호스트>:<포트>/<경로>:<파라미터>

2.5 스킴의 바다

- ✓ rtsp, rtspu
  - 실시간 스트리밍 프로토콜을 통해 읽을 수 있는 오디오 및 비디오와 같은 미디어 리소스 식별자
  - rtspu 스킴의 ‘u’는 udp 프로토콜이 사용됨을 의미
  - 기본형식
    rtsp://<사용자이름>:<비밀번호>@<호스트>:<포트>/<경로>
    rtspu://<사용자이름>:<비밀번호>@<호스트>:<포트>/<경로>

2.5 스킴의 바다

- ✓ file
  - 주어진 호스트기기 (로컬디스크, 네트워크 파일 시스템 등)에서 바로 접근할 수 있는 파일들을 나타냄
  - 각 필드도 일반적인 URL포맷을 사용
  - 호스트가 생략되어 있으면 기기의 로컬호스트가 기본값
  - 기본형식
    file://<호스트>/<경로>

2.5 스킴의 바다

- ✓ news
  - RFC1036에 정의된 바와 같이 특정 문서나 뉴스그룹에 접근하는데 사용
  - 리소스의 위치 정보를 충분히 포함하지 않는 특이한 속성이 있음
  - 뉴스 리소스는 여러 서버를 통하여 접근 가능하므로 위치에 독립적
  - '@' 문자는 뉴스 그룹을 가리키는 뉴스 URL과 특정 뉴스 문서를 가리키는 뉴스 URL을 구분하기 위해 사용
  - 기본형식
    news:<newsgroup>
    news:<news-article-id>

2.5 스킴의 바다

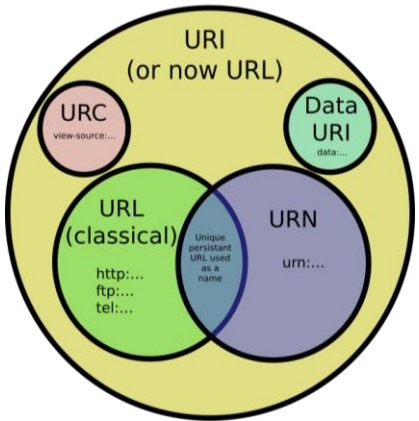
- ✓ telnet
  - 대화형 서비스에 접근하는데 사용
  - telnet URL자체가 객체를 가리키지는 않음
  - 리소스라고 할 수 있는 대화형 애플리케이션에 접속하는데 사용
  - 기본형식
    telnet://<사용자 이름>:<비밀번호>@<호스트>:<포트>/

2.6 미래

- ✓ URL의 장점과 단점
  - 장점
    - 모든 객체에 이름을 지을 수 있음
    - 새로운 포맷을 쉽게 추가할 수 있음
    - 인터넷 프로토콜 간의 일관된 작명 규칙을 제공
  - 단점
    - 특정 시점의 위치를 알려주는 역할을 하므로
      리소스가 옮겨지면 더는 사용할 수 없음

2.6 미래

- ✓ URN(Uniform Resource Name) 등장
  - URL의 단점을 극복하기 위해 등장
  - URN은 객체가 이동해도 항상 객체를 가리킬 수 있는 이름 제공
  - 지속 통합 자원 지시자(Persistent Uniform Resource Locator, PURL)를 사용하면 URL로 URN의 기능을 제공할 수 있음



참고

- HTTP 완벽가이드
- https://www.betterweb.or.kr/blog/url%EC%9D%B4%EB%9E%80/
- https://yurimkoo.github.io/http/2019/08/08/http-the-definitive-guide-1-2.html
- https://watrv41.gitbook.io/devbook/web/http/2-url
- https://hanamon.kr/%EB%84%A4%ED%8A%B8%EC%9B%8C%ED%81%AC-%EA%B8%B0%EB%B3%B8-url-uri-urn-%EC%B0%A8%EC%9D%B4%EC%A0%90/
- https://velog.io/tags/HTTP-%EC%99%84%EB%B2%BD-%EA%B0%80%EC%9D%B4%EB%93%9C
- https://docs.google.com/presentation/d/1VVFdDZQlV3W3v6cYDFUf_-oXqpB1lnQRP7t8hOwxP4M/edit#slide=id.g5e36212ad0_2_31
- https://programming119.tistory.com/194

QnA