



# HTTP 완벽 가이드

2장 URL과 리소스 @siyoung



## 2장에서 배우게 될 내용

URL 소개

URL 문법

상대URL, 단축 URL

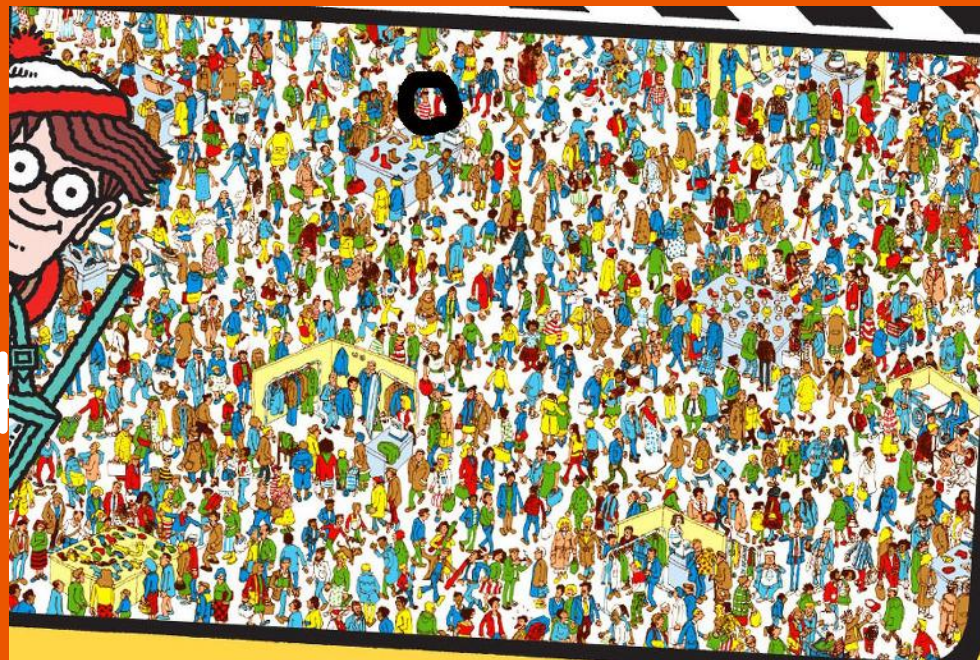
URL인코딩과 문자규칙

공통 URL 스킴

URL의 미래

인터넷상의  
리소스에는  
어떻게 접근해야  
할까?

---



---

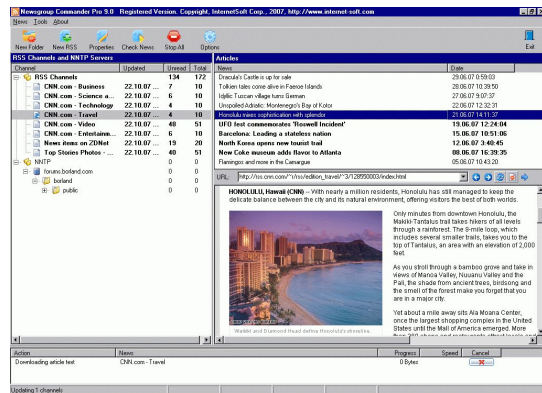
리소스에 주소를 붙이자.

URL

Uniform Resource Locator

# URL 소개 - URL이 없던 시절

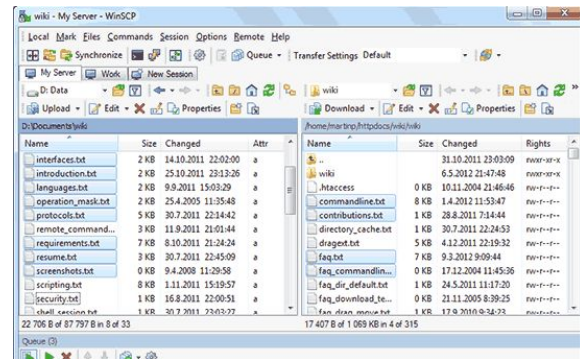
URL이 없던 시절..



뉴스 읽기



커뮤니티

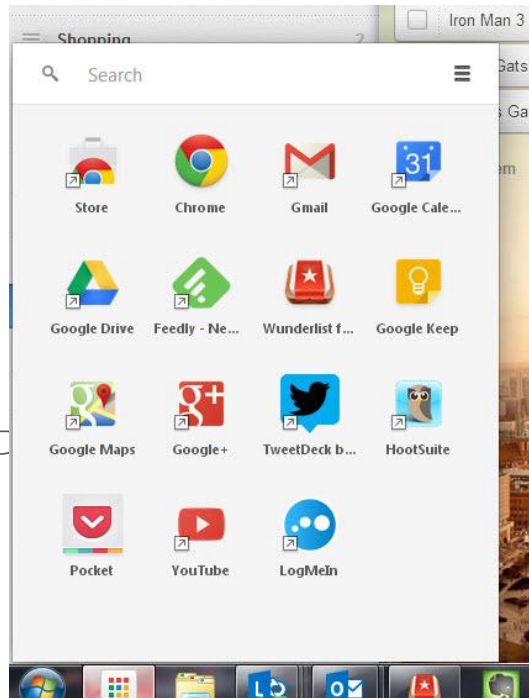


자료 받기

# URL 소개

URL을 사용하면

웹 브라우저를 사용해 위의 모든 리소스에 편리하게 접근 할 수 있다



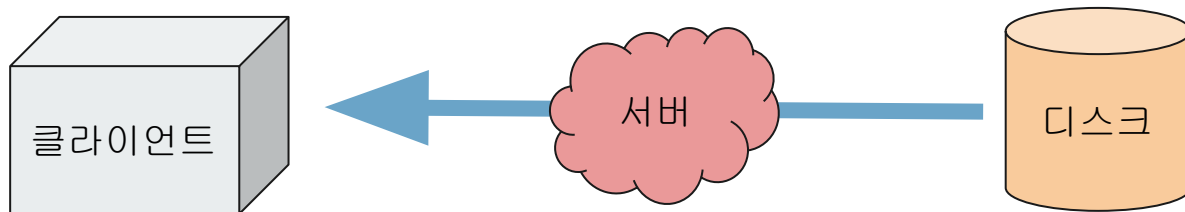
## URL 문법 - URL의 구조

스킴(어떻게) 호스트(어디에) 경로(무엇을)

<https://sports.news.naver.com/esports/news/read.nhn>

<ftp://ftp.lots-o-books.com/pub/complete-price-list.xls>

<mailto:president@whitehouse.gov>





# URL 문법 - 기본 형태

URL의 기본 형태

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

1. 세부적인 형태는 스킴(프로토콜)에 따라 달라진다.
2. 모든 컴포넌트를 가지는 URL 은 거의 없으며 스킴, 호스트, 경로가 중요

예) 구글에서 movie를 검색 했을 때

`https://www.google.com/search?q=movie`





# URL 문법 - 스킴

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

사용할 프로토콜(통신 규약) 정보

주어진 정보에 어떻게 접근할 것인가?

프로토콜을 정의 (http, ftp, gopher, mail ...)

대소문자를 구분하지 않음 (http = HTTP)



## URL 문법 - 호스트와 포트

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

리소스를 호스팅하는 장비와 장비 내의 리소스를 제공하는 서버의 주소

포트 컴포넌트는 서버가 열어놓은 네트워크 포트 (일반적으로 http는 80 사용)

http://www.joes-hardware.com:80/index.html

http://161.58.228.45:80/index.html



## URL 문법 - 사용자 이름과 비밀번호

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

데이터 접근을 위해 사용자 이름과 비밀번호를 요구하는 경우

FTP서버가 많이 사용함

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

ftp://**joe:joepassword**@www.joes-hardware.com/sales\_info.txt



# URL 문법 - 경로

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

리소스가 서버의 어디에 있는지 알려준다.

파일 시스템 경로와 유사한 구조

서버가 리소스의 위치를 찾는데 사용하는 정보

각 경로 조각은 자체만의 파라미터 컴포넌트를 가질 수 있음

[https://ko.wikipedia.org/wiki/통신\\_프로토콜](https://ko.wikipedia.org/wiki/통신_프로토콜)



# URL 문법 - 파라미터

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

URL 기본 컴포넌트들만으로 알 수 없는 정보 전달

리소스에 접근하기 위해 필요한 추가정보

경로별 설정 가능

ftp://prep.ai.mit.edu/pub/gnu;type=d

http://www.example.com/hammers;sale=false/index.html;graphics=true



## URL 문법 - 질의 (query)


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

주로 데이터베이스가 요청받을 리소스 형식의 범위를 좁히기 위해 사용

데이터베이스 게이트웨이로 전달되는 정보

&로 구분되는 key / value 쌍이 일반적인 형식

<https://www.google.com/search?ei=f6hFXdXkJOismAWEsKmgDg&q=apple>



# URL 문법 - 프래그먼트

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

HTML 리소스 내의 특정 위치를 가리킴

서버로 전달되지 않고 브라우저 내부에서 처리됨

<http://www.joes-hardware.com/tools.html#drills>

같은 서비스 내에서  
페이지만 이동하고 싶은데  
모든 URL를 다시 입력해야 할까?

—



---

단축 URL을 사용하자.  
기저 URL - 상대 URL





## 단축 URL

상대 URL - 리소스 내부에 있는 리소스를 간결하게 기술하는데 사용

자동 확장 - 많은 브라우저가 URL 일부를 입력하면 나머지 부분을 자동으로 입력하는 기능을 갖고 있음



## 단축 URL - 상대 URL

상대 URL - 현재 기저(base) URL을 바탕으로 한 URL

프래그먼트이거나 URL 일부

문서 집합의 위치를 변경하더라도 새로운 기저 URL에 의해 해석되므로 위치를 변경해도 잘 동작함

리소스 집합을 쉽게 변경 가능

ex) <http://www.samplepage/drills.html> → [./hammers.html](http://www.samplepage/hammers.html) → <http://www.samplepage/hammers.html>



# 단축 URL - 기저 URL

기저URL을 가져오는 방법

리소스에서 명시적으로 제공 - HTML의 경우 <BASE> 태그 사용해서 정의

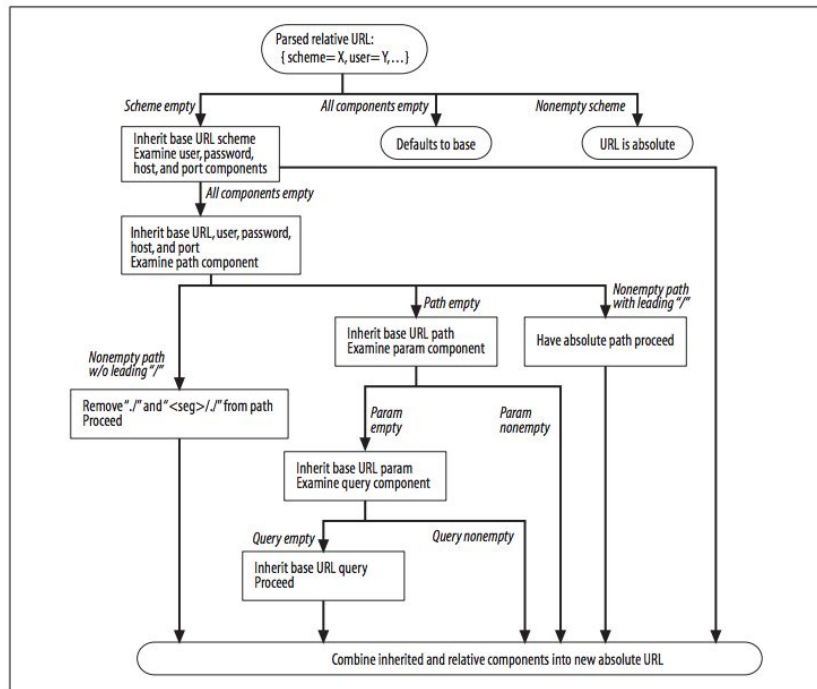
리소스를 포함하고 있는 기저URL - 해당 리소스의 URL을 기저URL로 사용

기저 URL이 없는 경우 - 절대 URL만으로 이루어져 있는 경우

# 단축 URL - 상대참조 해석하

URL 파싱 알고리즘 - 상대 URL을 절대 경로 형태로 변환

RFC1808에서 최초 기술, 이후 RFC2396에 포함





## 단축 URL - 상대참조 해석하기

사용 예제

`http://www.samplepage/drills.html` → `./hammers.html` → `http://www.samplepage/hammers.html`

경로는 `./hammers.html`, 기저 URL은 `http://www.samplepage/drills.html`

스킴은 비어있으므로 기저 URL의 스킴을 상속받는다(HTTP)

호스트와 포트 컴포넌트를 상속받아 새로운 절대 URL `http://www.samplepage/hammers.html` 을 만든다,



## 단축 URL - URL 확장

어떤 브라우저들은 자동으로 URL을 확장해준다. - URL을 빠르게 입력 가능

호스트명 확장

단순한 휴리스틱(어림짐작) 만으로 전체 URL을 완성해 준다.

yahoo → [www.yahoo.com](http://www.yahoo.com)

간편하지만 다른 HTTP애플리케이션에 문제를 야기하는 경우도 있음 (6장)



## 단축 URL - URL 확장

히스토리 확장

사용자가 과거에 방문했던 URL기록에서 선택할 수 있도록

전체 URL을 입력하는 대신 선택하면 됨

proxy를 사용하는 경우 다르게 작동할 수 있음 (6장에서 다름)

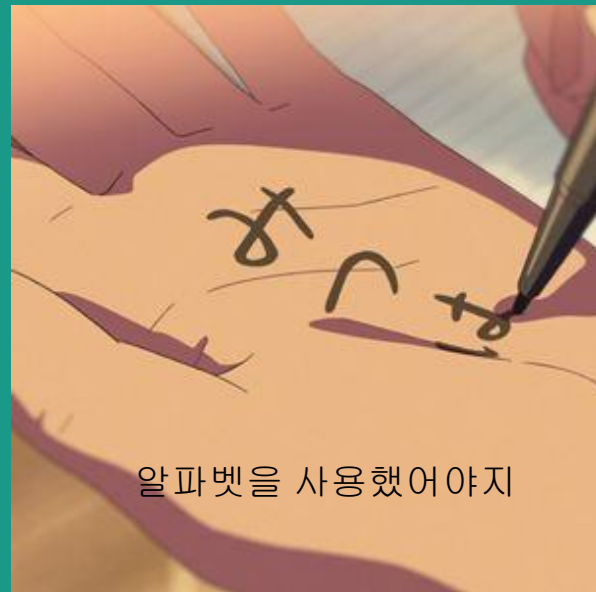


어떤 프로토콜을 사용하더라도  
안정성을 보장하는 이름은  
없을까?

—

---

리소스의 이름에는  
어떤 경우에도  
안전한 문자만  
사용하도록



알파벳을 사용했어야지



## URL 인코딩과 문자규칙 - 안전하지 않은 문자

URL은 프로토콜에 관계없이 정보가 안전하게 전달될 수 있도록 사용할 수 있는 문자를 제한

SMTP같은 프로토콜은 특정 문자를 제거할 수도 있다. (메시지에 7비트 인코딩을 사용)

URL에는 제거되지 않는 안전한 알파벳 문자들만 허용

안전하지 않은 문자는 이스케이프를 사용, 안전한 문자로 인코딩 할 수 있게



# URL 인코딩과 문자규칙 - 인코딩 체계

US-ASCII 문자 집합 - 적은 수 만을 포함

한계를 넘기 위해 이스케이프 문자를 이용한 인코딩 사용

문자	ASCII 코드	URL 예
~	126(0x7E)	<a href="http://www.joes.com/%7Ejoe">http://www.joes.com/%7Ejoe</a>
%	37(0x25)	<a href="http://www.joes.com/100%25tools.com">http://www.joes.com/100%25tools.com</a>



## URL 인코딩과 문자규칙 - 문자 제한

몇몇 문자는 URL내에서 특별한 의미로 예약되어 있다.

예약된 문자를 본래 목적이 아닌 다른 용도로 사용하려면 그 전에 반드시 인코딩해야 하는 문자들이 있음

문자	선점 및 제한
%	인코딩된 문자에 사용할 이스케이프 토큰으로 선점
/	경로 컴포넌트에 있는 경로 세그먼트를 나누는 용도로 선점
.	경로 컴포넌트에서 선점
..	경로 컴포넌트에서 선점



## URL 인코딩과 문자규칙 - 문자 제한

문자	선점 및 제한
#	프래그먼트의 구획문자로 선점
?	질의 문자열의 구획문자로 선점
:	파라미터의 구획문자로 선점
:	스킴, 사용자이름/비밀번호, 호스트/포트의 구획문자로 선점
\$\$, +	선점
@&=	특정 스킴에서 특별한 의미가 있기 때문에 선점
{ }   \ . ~ [ ] `	게이트웨이와 같은 여러 전송 에이전트에서 불안전하게 다룸



# URL 인코딩과 문자규칙 - 좀더 알아보기

안전하지 않은 문자를 URL에 사용하는 경우도 있지만,

안전하지 않은 문자를 이스케이프를 사용해 인코딩하지 않는것은 실수

인코딩할 문자를 결정하는 것은 브라우저단계에서부터하는것이 좋다.

본적은 있는것  
같은데 이름은  
잘 몰라요..

---





---

# 자주쓰는 스킴에 대해 확실히 알아보자



조유리

김채원

최예나



# 공통 URL 스킴 - http

Hypertext Transfer Protocol

사용자이름 / 비번 외에 모든 컴포넌트를 사용

포트값이 생략되어 있으면 기본값은 80

기본형식

<http://<호스트>:<포트>/<경로>?<질의>#<프래그먼트>>



# 공통 URL 스킴 - https

HTTP 스킴과 같다.

HTTP 커넥션의 양 끝단에서 넷스케이프에서 개발한 보안 소켓 계층(Secure Sockets Layer, SSL) 사용, 암호화

기본 포트값은 443

기본형식

<https://<호스트>:<포트>/<경로>?<질의>#<프래그먼트>>



# 공통 URL 스킴 - mailto

이메일 주소를 가리킴

다른 스킴과는 다르게 동작하므로, 표준 URL과 다른 포맷을 가짐

문법은 RFC 822에 기술되어 있음 ([RFC란?](#))

기본형식

<mailto:example@gmail.com>



# 공통 URL 스킴 - ftp

파일 전송 프로토콜(File Transfer Protocol)

FTP서버의 파일을 내려받거나 콘텐츠 목록을 가져올때 사용

FTP는 URL이 등장하기 전부터 있었음

기본형식

`ftp://<사용자이름>:<비밀번호>@<호스트>:<포트>/<경로>:<파라미터>`



## 공통 URL 스킴 - rtsp, rtspu

실시간 스트리밍 프로토콜(Real Time Streaming Protocol)을 통해 읽을 수 있는 오디오와 비디오

미디어 리소스 식별자

rtspu스킴의 'u'는 udp프로토콜이 사용됨을 의미

기본형식

rtsp://<사용자이름>:<비밀번호>@<호스트>:<포트>/<경로>



## 공통 URL 스킴 - 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>`



## 공통 URL 스킴 - news

RFC1036에 정의된 바와 같이 특정 문서나 뉴스그룹에 접근하는데 사용

리소스의 위치 정보를 충분히 포함하지 않는 특이한 속성이 있다.

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>`





## 공통 URL 스킴 - telnet

telnet 대화형 서비스에 접근하는데 사용

telnet URL 자체가 객체를 가리키지는 않음

리소스라고 할 수 있는 대화형 애플리케이션에 접속하는데 사용

기본형식

`telnet://<사용자 이름>:<비밀번호>@<호스트>:<포트>/`

예) `telnet://t2tmud.org:9999`

# URL의 미래는 어떤 모습일까?

---



---

# URN Uniform Resource Names



# URL의 미래

서버 리소스 이름은 통합 자원 식별자(uniform resource identifier) 혹은 URI라고 불린다.

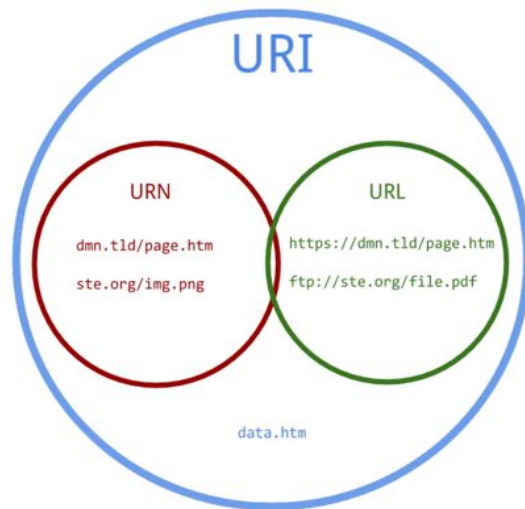
URL을 주소이고 실제 이름은 아님. 즉 리소스가 옮겨지면 더는 사용할 수

없다?

위치와 상관없이 객체의 실제 이름을 사용하는 것

인터넷 기술 태스크포스(IETF)는 URN이라는 새로운 표준 작업에 착수

객체가 옮겨지더라도 항상 객체를 가리키는 이름을 제공





# URL의 미래

지속 통합 자원 지시자(Persistent uniform resource locators, PURL)을 사용하면

URL로 URN의 기능을 제공할 수 있음

클라이언트는 위치 할당자에게 영구적인 URL을 요청할 수 있으며

영구적인 URL은 클라이언트를 리소스의 실제 URL로 연결해 준다.



# URL의 미래

주소체계를 바꾸는 것은 매우 큰작업이며, URL은 당분간 계속 사용할것

언젠가는 URN이 URL을 대체할 수 있다.



감사합니다.