

---

# HTML에서 웹앱까지

## 10주차\_01

한 동 대 학 교  
김군오 교수

# 학습 목표: AJAX 기술 소개 및 활용하기

---

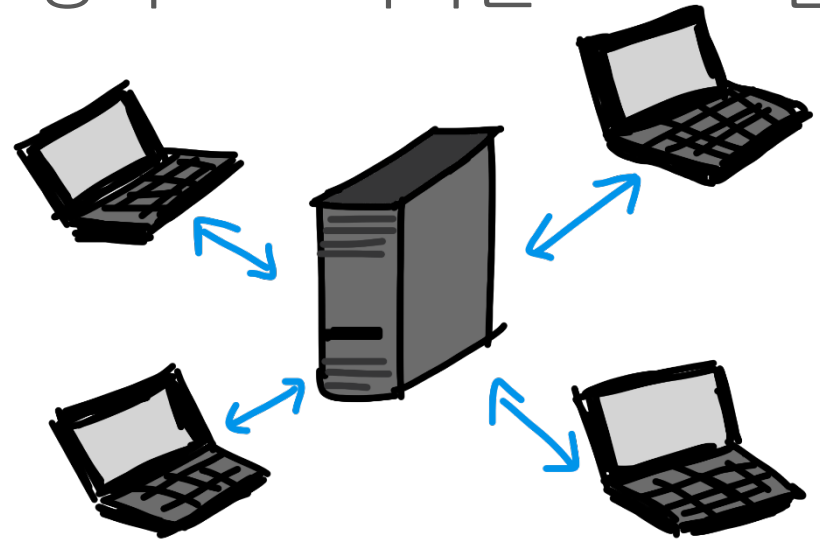
## 학습내용:

- 웹 서버와 브라우저 통신 방법
- http 프로토콜
- AJAX 기술 소개
- JQuery AJAX 활용

# 서버 – 클라이언트 모델

---

- 네트워크를 이용하는 어플리케이션을 만들 때 사용
- 서버: 클라이언트의 요청에 따라 파일이나 데이터를 보내주는 프로그램
- 클라이언트: 서버에 파일이나 데이터를 요청하고 소비하는 프로그램

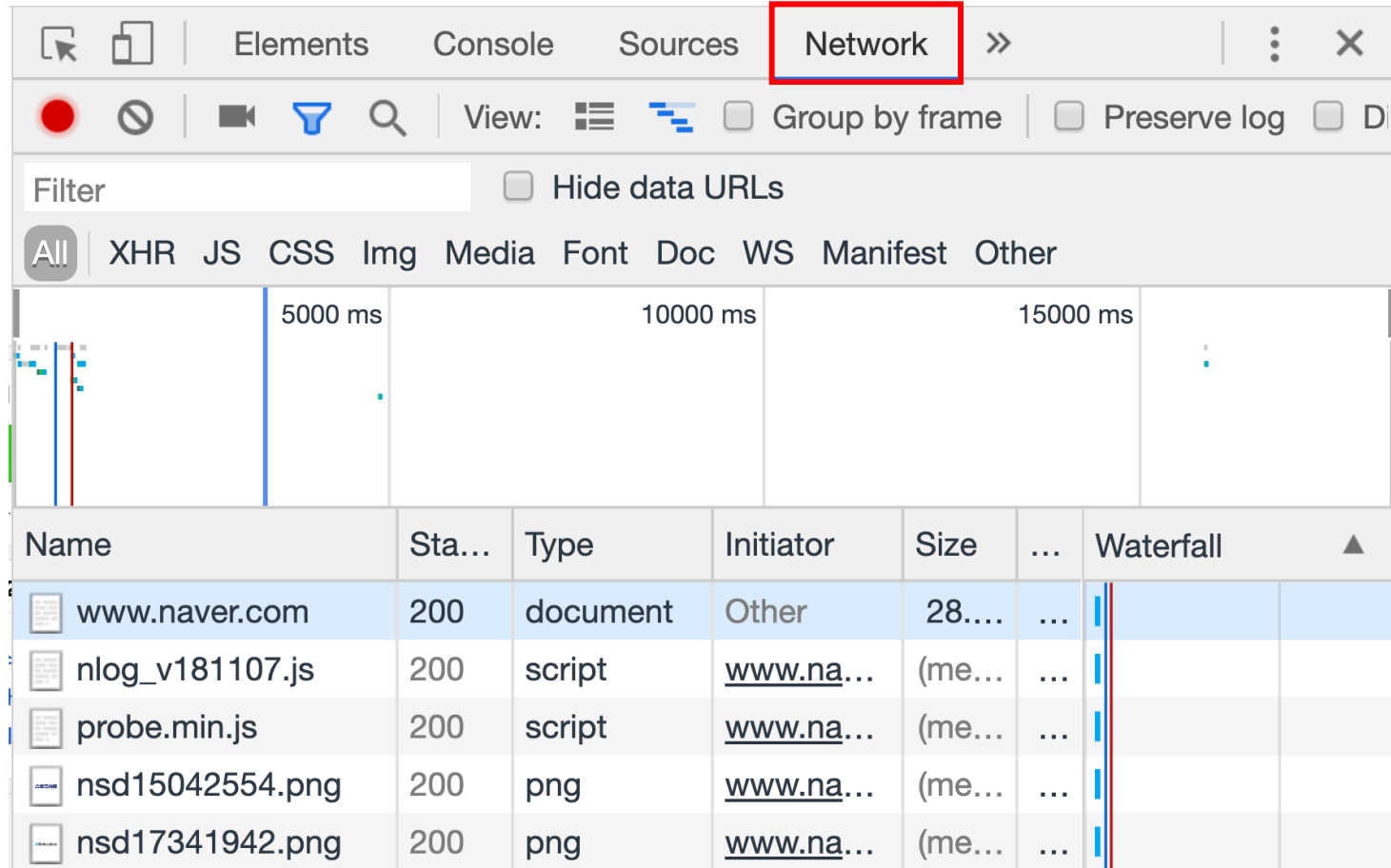


# 웹 브라우저

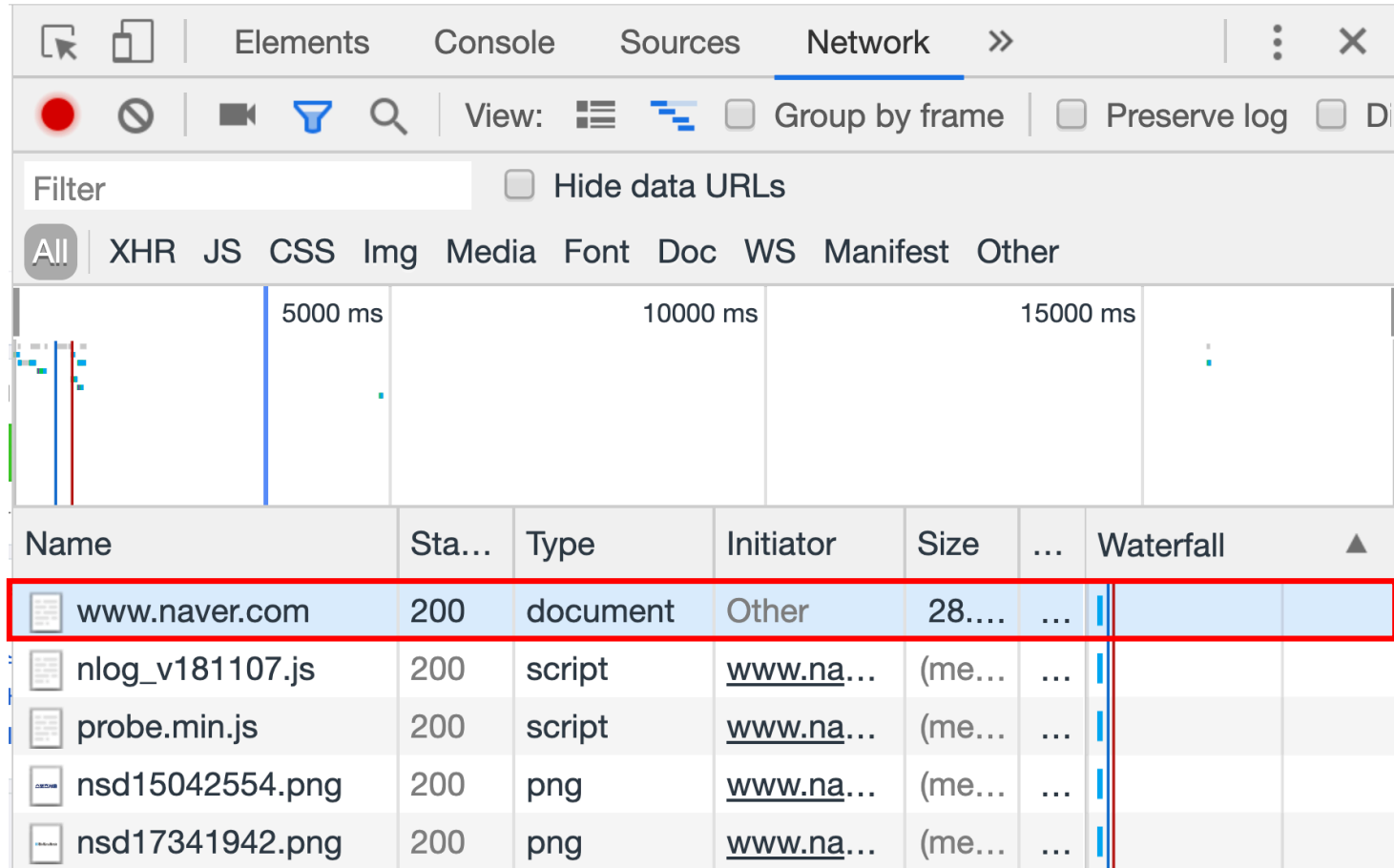
---

- 웹 서버에 HTML 파일을 요청해 다운로드하고 화면에 표시하는 프로그램
- HTML 을 분석해 화면에 그리고 이미지나 스트립트 파일 등 추가적인 리소스가 필요하면 서버에 요청한다

# 웹 브라우저



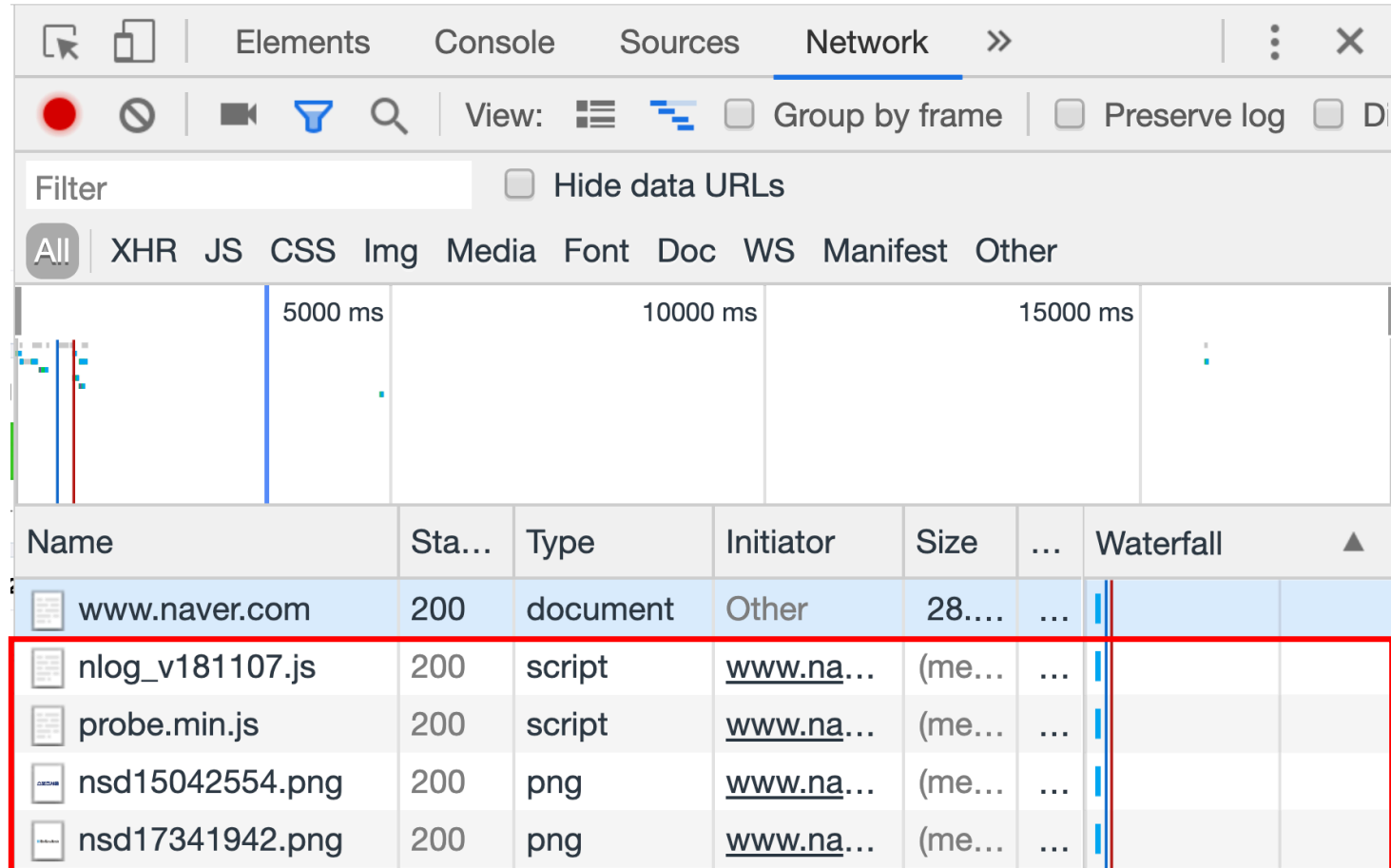
# 웹 브라우저



The screenshot displays the Network tab of a web browser's developer tools. The top section includes tabs for Elements, Console, Sources, and Network. Below these are various icons and a filter input field. The main area shows a list of network requests. The first request, 'www.naver.com', is highlighted with a red box. The table below provides details for each request.

Name	Sta...	Type	Initiator	Size	...	Waterfall
www.naver.com	200	document	Other	28....	...	
nlog_v181107.js	200	script	www.na...	(me...	...	
probe.min.js	200	script	www.na...	(me...	...	
nsd15042554.png	200	png	www.na...	(me...	...	
nsd17341942.png	200	png	www.na...	(me...	...	

# 웹 브라우저



The screenshot shows the Network tab of a web browser's developer tools. The top bar includes tabs for Elements, Console, Sources, and Network. Below the tabs are various icons and a search bar. The main area displays a list of network requests. The first request is 'www.naver.com' (document). The subsequent four requests are highlighted with a red box: 'nlog\_v181107.js' (script), 'probe.min.js' (script), 'nsd15042554.png' (png), and 'nsd17341942.png' (png). All these requests have a status of 200 and were initiated by 'www.na...'. The table columns are Name, Sta..., Type, Initiator, Size, and Waterfall.

Name	Sta...	Type	Initiator	Size	...	Waterfall
www.naver.com	200	document	Other	28....	...	
nlog_v181107.js	200	script	www.na...	(me...	...	
probe.min.js	200	script	www.na...	(me...	...	
nsd15042554.png	200	png	www.na...	(me...	...	
nsd17341942.png	200	png	www.na...	(me...	...	

# HTTP 프로토콜

- 웹 서버와 웹 브라우저가 서로 통신하는 규칙
- 요청과 응답으로 이루어짐
- 최근에는 웹 브라우저 뿐만 아니라 컴퓨터 프로그램, 스마트폰 앱 등에서 널리 사용되고 있음

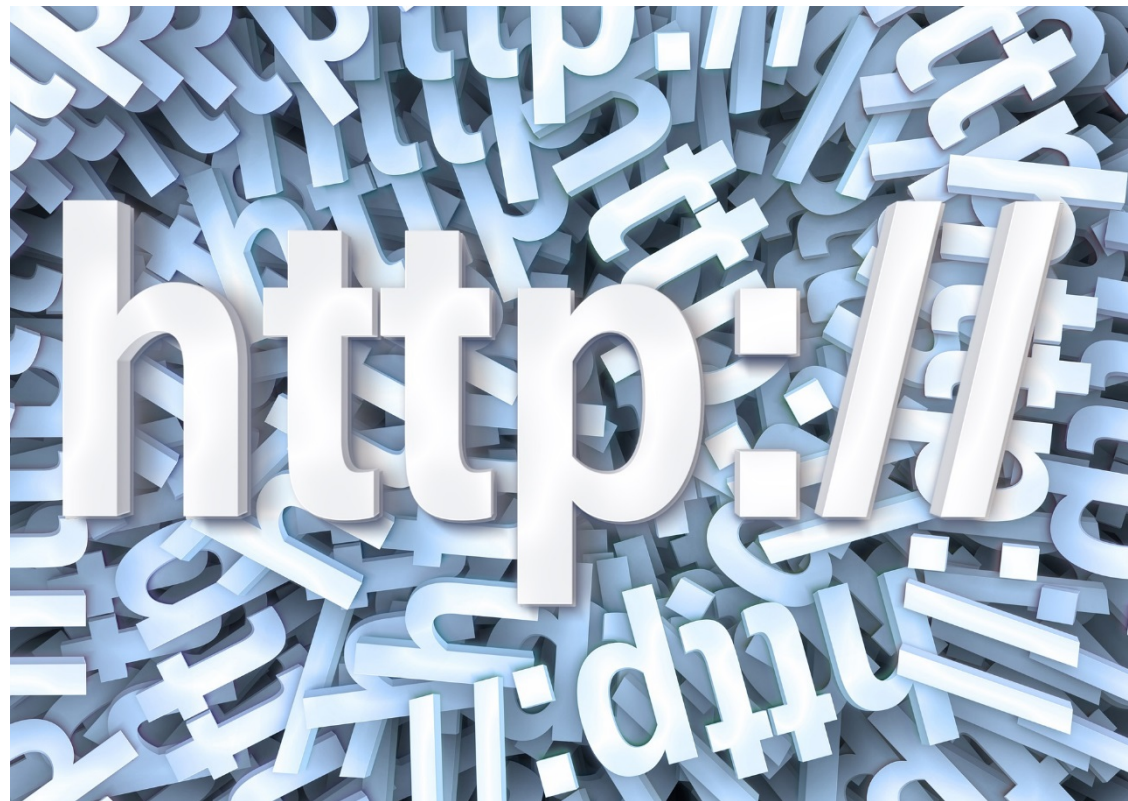


# HTTP 요청

- GET : 정보를 요청하기 위해서 사용한다. (SELECT)
- POST : 정보를 추가하기 위해서 사용한다. (INSERT)
- PUT : 정보를 업데이트하기 위해서 사용한다. (UPDATE)
- DELETE : 정보를 삭제하기 위해서 사용한다. (DELETE)

# HTTP 응답

- 2xx : 성공
- 3xx : 리디렉션
- 4xx : 요청 오류
- 5xx : 서버 오류



# AJAX

---

- Asynchronous JavaScript and XML
- 페이지를 로드 하지 않고 브라우저에서 서버에 요청할 수 있는 기법
- HTML, 텍스트, JSON 등의 데이터를 주고 받음
- 비동기 실행: 데이터를 로드 하는 동안 자바스크립트 코드를 중단하지 않고 계속 실행됨

# Jquery AJAX

- Jquery 를 이용하면 간편하게 ajax 통신을 할 수 있음
- 형식

`$.ajax(url, options)`

url : 요청하려는 웹 리소스의 url

options : 요청에 대한 옵션

- done, fail, always 함수를 이용하여 비동기 작업이 완료되었을 때 수행할 함수 지정

# AJAX 실습

```
<script src="https://code.jquery.com/jquery-3.4.1.min.js"></script>
<script>
    $.ajax('data.txt')
        .done(function(data) {
            // 성공시 실행할 코드
            console.log(data);
        })
        .fail(function(error) {
            // 실패시 실행할 코드
            console.log("에러발생!");
        })
        .always(function() {
            // 항상 실행할 코드
            console.log("ajax 호출이 완료되면 실행됩니다.");
        });

    console.log("AJAX 호출은 비동기 방식입니다.");
</script>
```

# AJAX 실습

```
<script src="https://code.jquery.com/jquery-3.4.1.min.js"></script>
<script>
    $.ajax('data.txt')
        .done(function(data) {
            // 성공시 실행할 코드
            console.log(data);
        })
        .fail(function(error) {
            // 실패시 실행할 코드
            console.log("에러발생!");
        })
        .always(function() {
            // 항상 실행할 코드
            console.log("ajax 호출이 완료되면 실행됩니다.");
        });

    console.log("AJAX 호출은 비동기 방식입니다.");
</script>
```

# AJAX 실습

```
<script src="https://code.jquery.com/jquery-3.4.1.min.js"></script>
<script>
    $.ajax('data.txt')
        .done(function(data) {
            // 성공시 실행할 코드
            console.log(data);
        })
        .fail(function(error) {
            // 실패시 실행할 코드
            console.log("에러발생!");
        })
        .always(function() {
            // 항상 실행할 코드
            console.log("ajax 호출이 완료되면 실행됩니다.");
        });

    console.log("AJAX 호출은 비동기 방식입니다.");
</script>
```

# AJAX 실습

```
<script src="https://code.jquery.com/jquery-3.4.1.min.js"></script>
<script>
    $.ajax('data.txt')
        .done(function(data) {
            // 성공시 실행할 코드
            console.log(data);
        })
        .fail(function(error) {
            // 실패시 실행할 코드
            console.log("에러발생!")
        })
        .always(function() {
            // 항상 실행할 코드
            console.log("ajax 호출이 완료되면 실행됩니다.");
        });

    console.log("AJAX 호출은 비동기 방식입니다.")
</script>
```



# AJAX 실습

```
<script src="https://code.jquery.com/jquery-3.4.1.min.js"></script>
<script>
    $.ajax('data.txt')
        .done(function(data) {
            // 성공시 실행할 코드
            console.log(data);
        })
        .fail(function(error) {
            // 실패시 실행할 코드
            console.log("에러발생!")
        })
        .always(function() {
            // 항상 실행할 코드
            console.log("ajax 호출이 완료되면 실행됩니다.");
        });

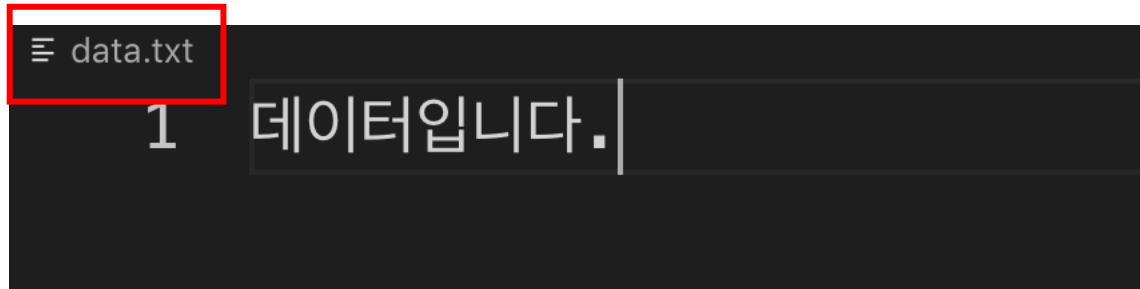
    console.log("AJAX 호출은 비동기 방식입니다.")
</script>
```

# AJAX 실습

```
<script src="https://code.jquery.com/jquery-3.4.1.min.js"></script>
<script>
    $.ajax('data.txt')
        .done(function(data) {
            // 성공시 실행할 코드
            console.log(data);
        })
        .fail(function(error) {
            // 실패시 실행할 코드
            console.log("에러발생!")
        })
        .always(function() {
            // 항상 실행할 코드
            console.log("ajax 호출이 완료되면 실행됩니다.");
        });

    console.log("AJAX 호출은 비동기 방식입니다.")
</script>
```

# AJAX 실습



# AJAX 실습

✖ ▶ Access to XMLHttpRequest at 'file:///Users/ma jquery-3.4.1.min.js:2  
c/Documents/repo/kmooc%20example/data.txt' from origin 'null' has  
been blocked by CORS policy: Cross origin requests are only supported  
for protocol schemes: http, data, chrome, chrome-extension, https.

에러발생!

ex.html:15

ajax 호출이 완료되면 실행됩니다.

ex.html:19

AJAX 호출은 비동기 방식입니다.

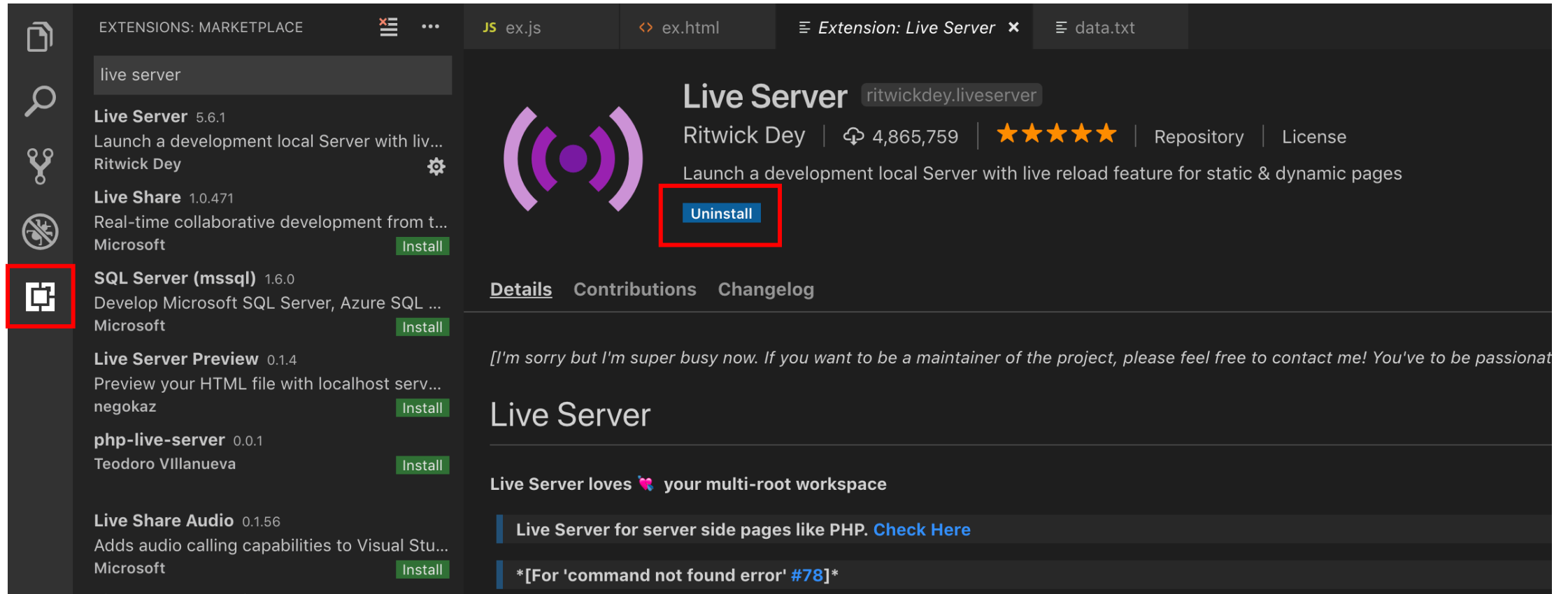
ex.html:22



# 개발용 웹 서버

- 로컬 환경 (개발용 PC) 에서 손쉽게 웹 서버 실행
- 현재 폴더 기준으로 URL 을 통해 파일 접근
- Live reload 기능 제공
  - 파일이 변경되는 것을 감지해서 코드를 수정하고 저장하는 것 만으로 페이지를 “새로 고침” 해 줌

# Live server 설치



The screenshot displays the Visual Studio Code interface with the Extensions Marketplace open. The search bar contains 'live server'. The extension 'Live Server' by Ritwick Dey is selected, showing its details. The 'Uninstall' button is highlighted with a red box. The left sidebar also shows a red box around the extension icon.

**EXTENSIONS: MARKETPLACE**

live server

**Live Server** 5.6.1  
Launch a development local Server with liv...  
Ritwick Dey

**Live Share** 1.0.471  
Real-time collaborative development from t...  
Microsoft [Install](#)

**SQL Server (mssql)** 1.6.0  
Develop Microsoft SQL Server, Azure SQL ...  
Microsoft [Install](#)

**Live Server Preview** 0.1.4  
Preview your HTML file with localhost serv...  
negokaz [Install](#)

**php-live-server** 0.0.1  
Teodoro Villanueva [Install](#)

**Live Share Audio** 0.1.56  
Adds audio calling capabilities to Visual Stu...  
Microsoft [Install](#)

**Live Server** ritwickdey.liveserver  
Ritwick Dey | 4,865,759 | ★★★★★ | Repository | License  
Launch a development local Server with live reload feature for static & dynamic pages

[Uninstall](#)

[Details](#) [Contributions](#) [Changelog](#)

[I'm sorry but I'm super busy now. If you want to be a maintainer of the project, please feel free to contact me! You've to be passionate]

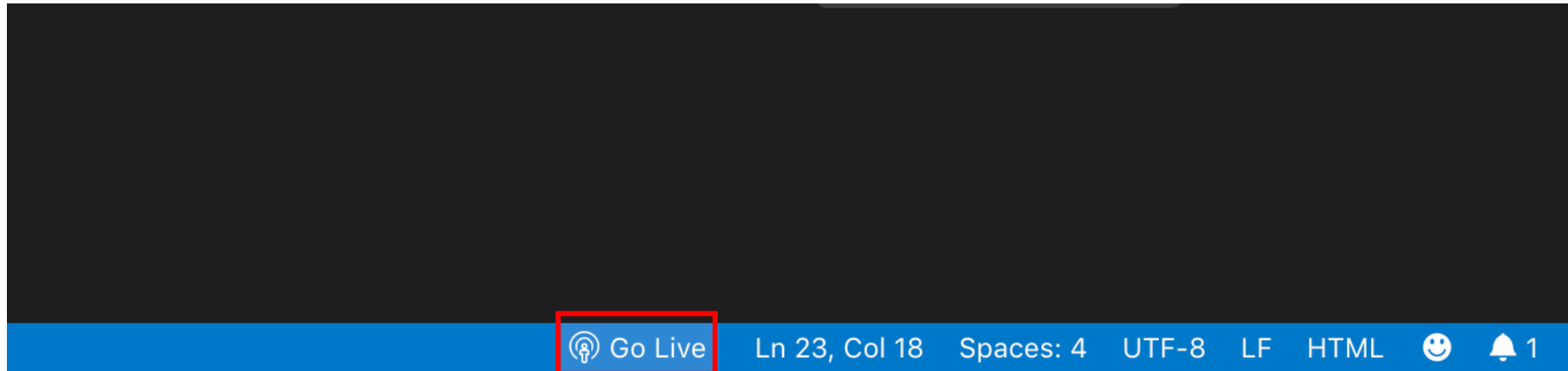
**Live Server**

Live Server loves ❤️ your multi-root workspace

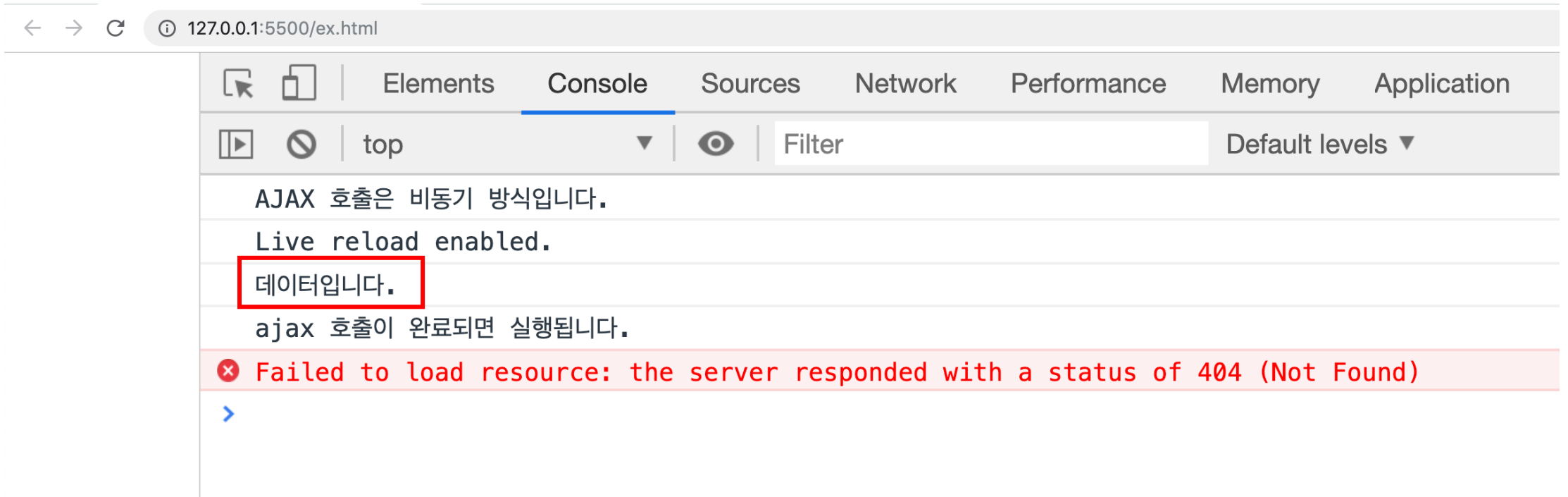
Live Server for server side pages like PHP. [Check Here](#)

\*[For 'command not found error' #78]\*

# Live server 실행

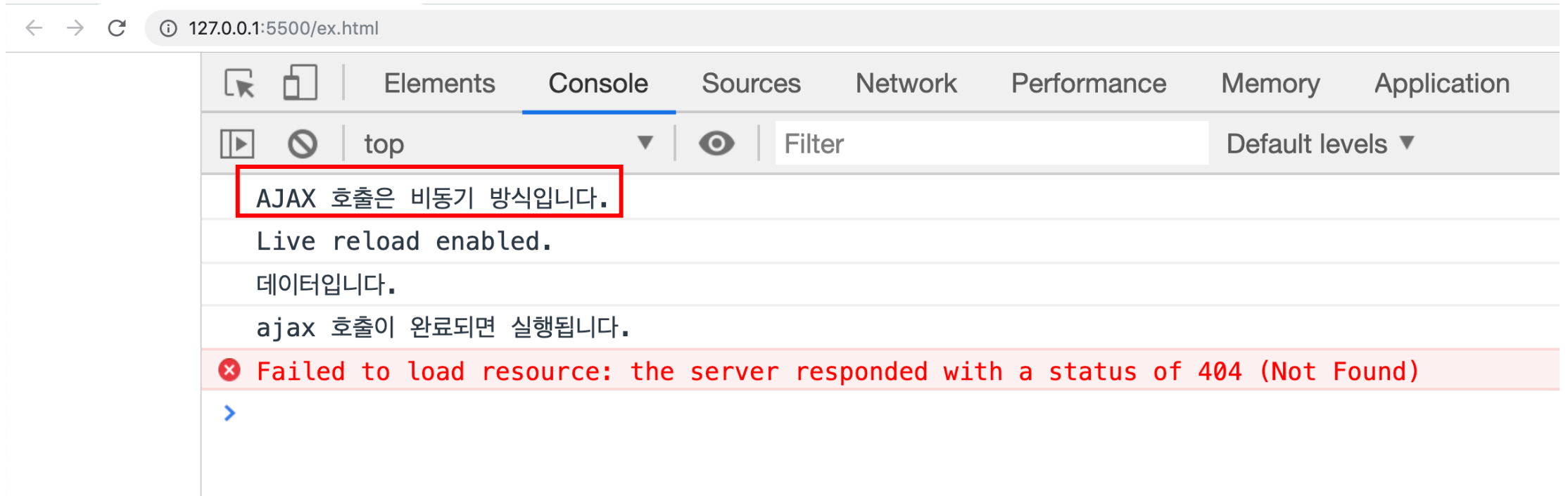


# AJAX 호출 결과 확인

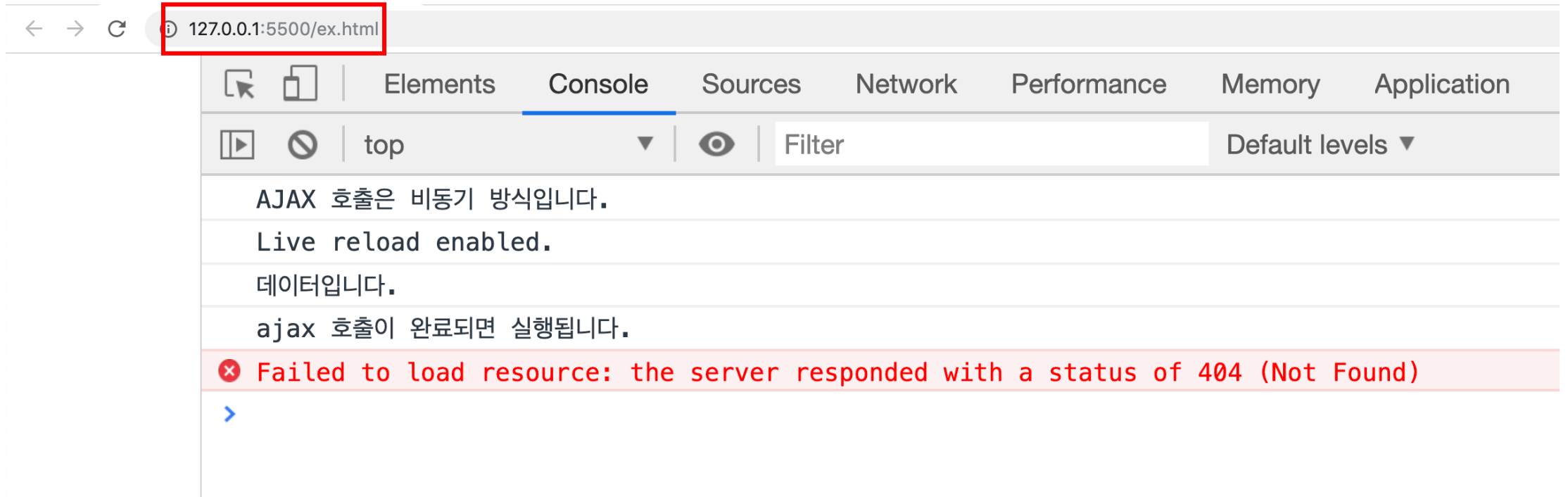




# AJAX 호출 결과 확인



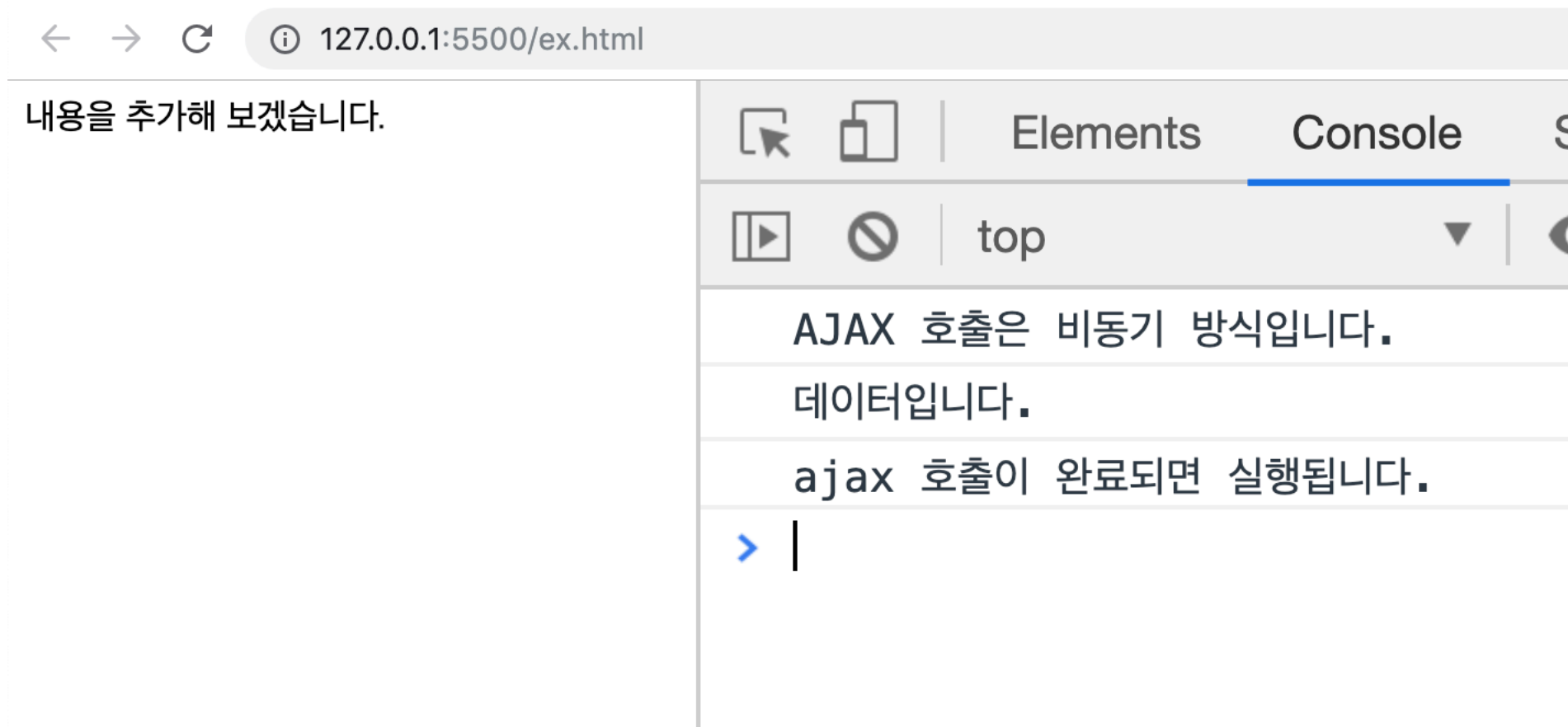
# AJAX 호출 결과 확인



# Live reload

```
<> ex.html • JS ex.js ≡ data.txt
<> ex.html ▶ ...
1  <!DOCTYPE html>
2  <html>
3      <body>
4          내용을 추가해 보겠습니다.
5          <script src="https://code.jquery.com"
6          <script>
7              $.ajax('data.txt')
8              .done(function(data) {
9                  // 성공 시 실행할 코드
```

# Live reload



# 사이드 메뉴 레이아웃

- 사이드 메뉴를 표시하는 HTML 문서
- 메뉴를 선택하면 해당 내용이 저장되어있는 다른 HTML 을 요청하고 화면에 표시

# 사이드 메뉴 레이아웃 : 파일 구조

```
<> 메뉴 1.html  
<> 메뉴 2.html  
<> 메뉴 3.html  
JS app.js  
<> index.html  
# style.css
```

```
<> 메뉴 1.html
```

```
1  첫번째 페이지 입니다.
```

```
<> 메뉴 2.html
```

```
1  두번째 페이지 입니다.
```

```
<> 메뉴 3.html
```

```
1  세번째 페이지 입니다.
```

# 사이드 메뉴 레이아웃 : HTML

```
1  <!DOCTYPE html>
2  <html>
3      <head>
4          <title>10-1</title>
5          <link rel="stylesheet" type="text/css" href="./style.css"></link>
6      </head>
7  +   <body> ...
21      </body>
22  </html>
```

# 사이드 메뉴 레이아웃 : HTML

```
7      <body>
8          <nav id="sidemenu">
9              <ul>
10                 <li>메뉴 1</li>
11                 <li>메뉴 2</li>
12                 <li>메뉴 3</li>
13             </ul>
14         </nav>
15         <div id="content">
16             보고싶은 메뉴를 클릭하세요.
17         </div>
18
19         <script src="https://code.jquery.com/jquery-3.4.1.min.js"></script>
20         <script src="app.js"></script>
21     </body>
```



# 사이드 메뉴 레이아웃 : CSS

```
1  #sidemenu, #content {  
2    |    float: left;  
3  }  
4  
5  #content {  
6    |    padding: 16px 40px;  
7  }
```

# 사이드 메뉴 레이아웃 : JAVASCRIPT

```
1  var menu = $('#sidemenu li');
2  var content = $('#content');
3
4  menu.click(function(event) {
5      var url = event.target.innerText + '.html';
6      $.ajax(url)
7          .done(function(html) {
8              content.html(html);
9          });
10 });
```

# 사이드 메뉴 레이아웃

- 메뉴 1      보고싶은 메뉴를 클릭하세요.
- 메뉴 2
- 메뉴 3

- 메뉴 1      두번째 페이지 입니다.
- 메뉴 2
- 메뉴 3

# \$.getJSON

- Jquery 의 getJSON 함수 이용
- .ajax 함수와 비슷하지만 데이터를 자바스크립트 객체 변환

# 단어장 만들기

- 단어장은 JSON 형식으로 저장되어 있음
- \$.getJSON 함수로 단어장 내용을 불러와 화면에 표시

# 단어장 만들기

```
{ } dict.json ▶ ...  
1  [  
2      {  
3          "title": "capture",  
4          "description": "포획하다, 사로잡다, ...을 붙잡다"  
5      },  
6      {  
7          "title": "captivate",  
8          "description": "...을 매혹하다, 마음을 사로잡다, 매혹시키다"  
9      },  
10     {  
11         "title": "captious",  
12         "description": "트집 잡는, 책망하는, 헐뜯는"  
13     }  
14 ]
```

# 단어장 만들기

```
<ul></ul>  
<h2></h2>  
<p></p>
```

# 단어장 만들기

13

```
var dict = $('ul');
```

14

```
var title = $('h2');
```

15

```
var p = $('p');
```



# 단어장 만들기

```
17 $.getJSON('dict.json')
18   .done(function(data) {
19       for (var word of data) {
20           var li = $('<li></li>');
21           li.text(word.title);
22           li.data('word', word);
23           li.click(function(event) {
24               var el = $(event.target);
25               var word = el.data('word');
26               title.text(word.title);
27               p.text(word.description);
28           });
29
30           dict.append(li);
31       }
32   });
```

# 단어장 만들기

- capture
- captivate
- captious

## capture

포획하다, 사로잡다, ...을 붙잡다

- capture
- captivate
- captious

## captivate

...을 매혹하다, 마음을 사로잡다, 매혹시키다

# 요약

- 웹 서버와 브라우저 통신 방법
- http 프로토콜
- AJAX 기술 소개
- JQuery AJAX 활용

# 차시 예고

---

- 10-2 : Open API 소개 및 활용하기
  - Open API
  - 한동대학교 API
  - 카카오 지도

---

**강의를 마치겠습니다**  
**수고하셨습니다**

10주차\_01 AJAX