M02-04 · GIBALJA VISUAL MANUAL

브라우저 개발자 도구로 API 찾기

> **한 문장 목표:** 화면에서 행동 하나를 실행하고, 그 행동과 함께 생긴 요청을 URL·메서드·입력·응답·호출 근거로 검증해 API 후보로 기록합니다.

난이도	개념	실습	셀프 테스트	최종 산출물
Level 1	50분	55분	15분	화면 행동·API 지도, API 후보 분석표, 안전한 증거 묶음

요청 이름을 보고 API를 맞히는 수업이 아닙니다. Network의 많은 줄에서 화면 행동과 함께 변한 요청을 찾고, 여섯 가지 증거가 서로 맞는지 확인하는 관찰 훈련입니다.

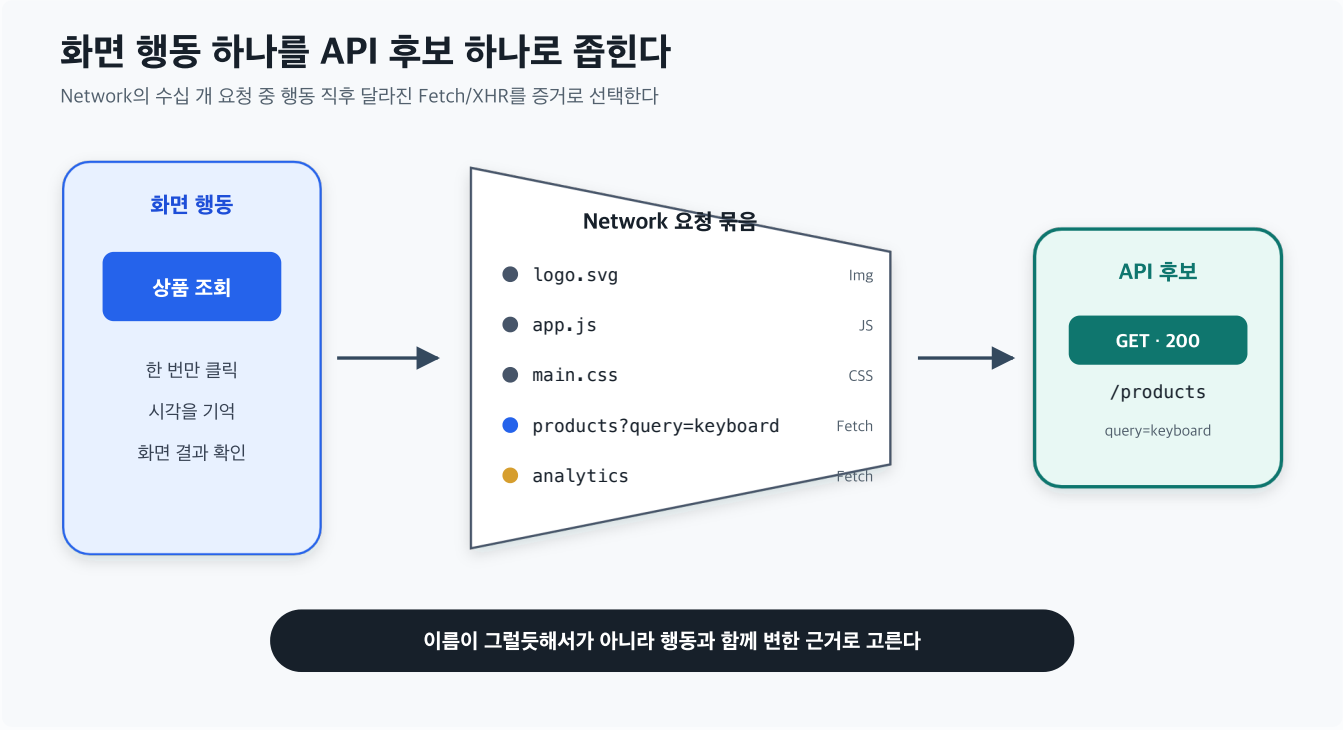
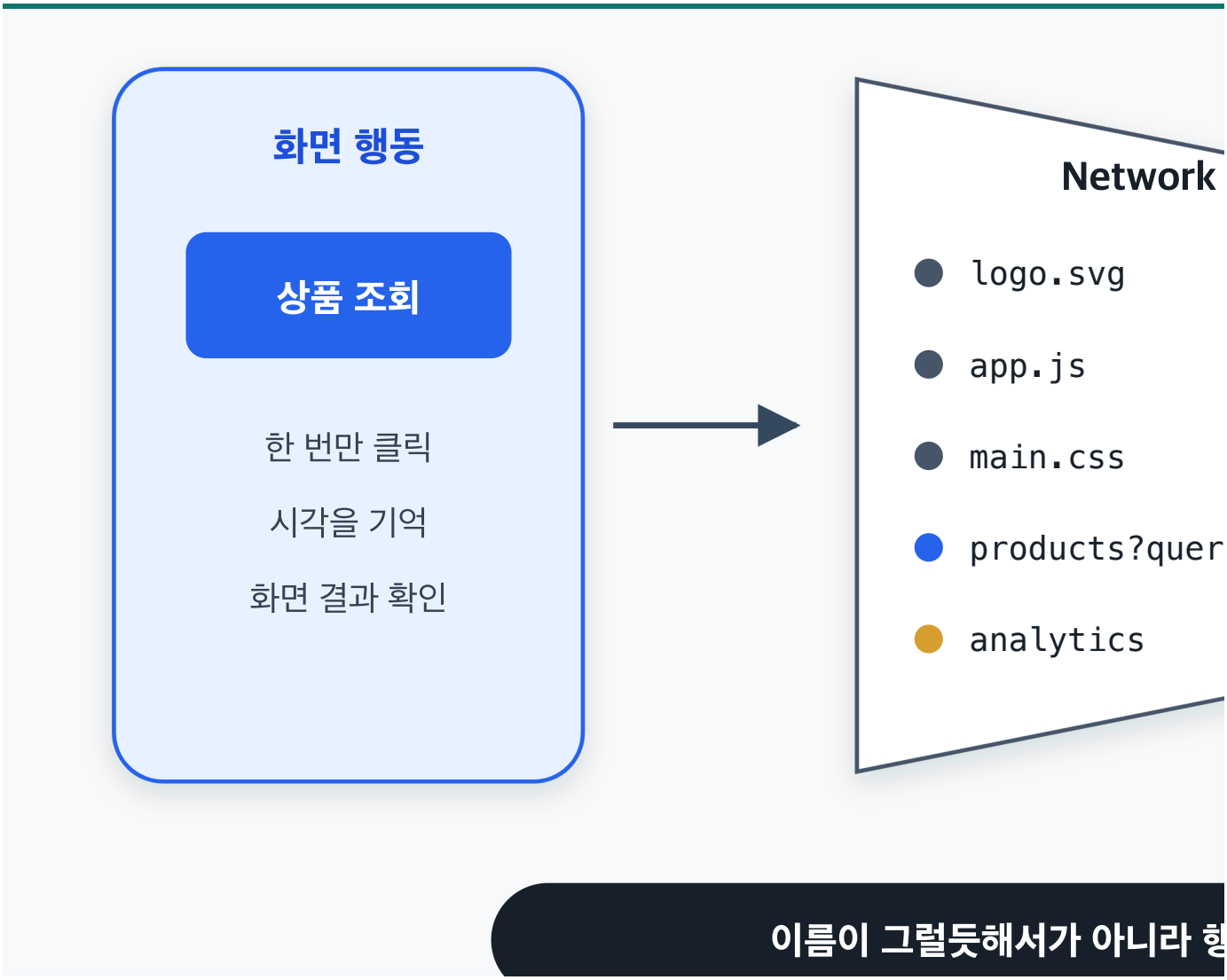
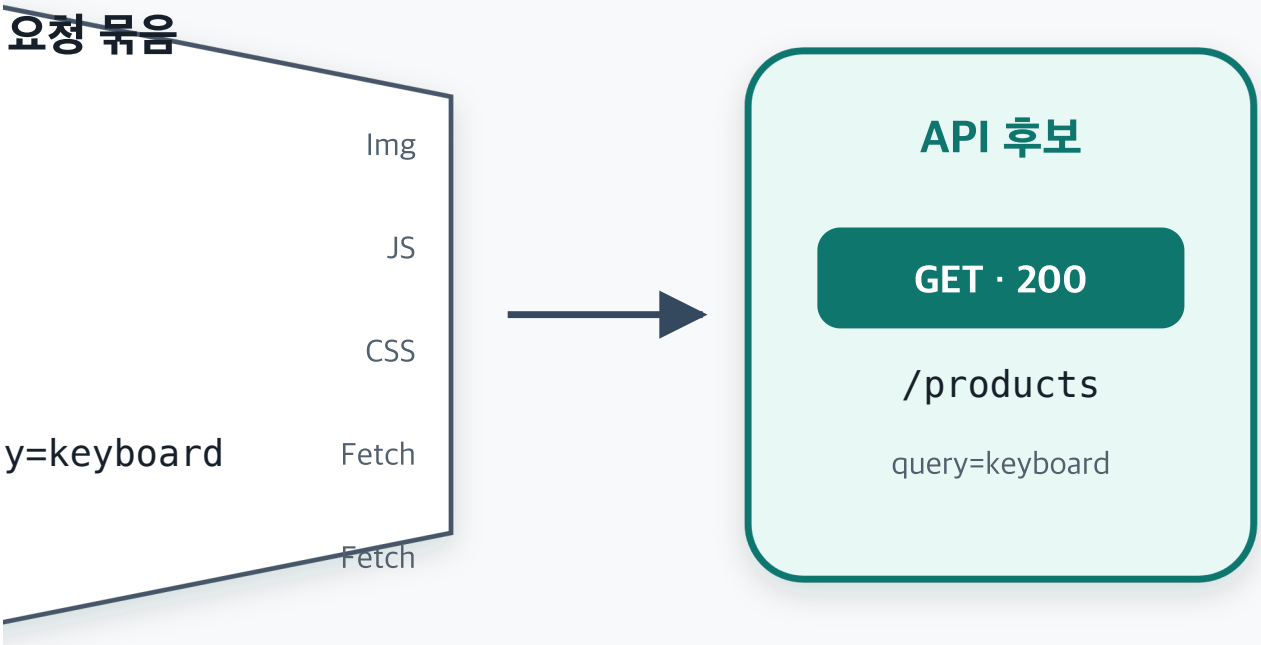


그림 1. 한 번의 화면 행동에서 시작하면 수십 개 요청을 재현 가능한 API 후보 하나로 좁힐 수 있습니다.


화면 행동
상품 조회
한 번만 클릭
시각을 기억
화면 결과 확인
Network
logo.svg
app.js
main.css
products?quer
analytics
이름이 그럴듯해서가 아니라 행



동과 함께 변한 근거로 고른다

1. 이 PDF를 공부하는 방법

1회차 · 그림만 훑기 · 15분

그림 1부터 그림 10까지 보고 `행동 → 필터 → 검증 → 기록` 흐름을 소리 내어 설명합니다. 모르는 용어는 뒤의 용어집에 표시만 하고 계속 갑니다.

2회차 · API 후보 찾기 · 25분

실습 화면을 열고 **상품 조회** 한 번만 실행합니다. Network에서 `Fetch/XHR`로 좁힌 뒤 `products` 요청의 메서드·URL·질의·응답을 기록합니다.

3회차 · POST와 OPTIONS 구분하기 · 40분

필터 적용을 실행합니다. 같은 URL에 나타난 `OPTIONS`와 `POST`를 짝으로 보고, 실제 업무 입력이 어느 요청에 담겼는지 찾습니다.

4회차 · 전달 가능한 분석서 만들기 · 25분

화면 행동·API 지도와 API 후보 분석표를 채우고, 민감정보를 제거한 뒤 동료에게 전달할 한 문장을 작성합니다.

5회차 · 셀프 테스트 · 15분

정답을 가리고 10문제를 풉니다. 틀린 문제는 관련 그림으로 돌아가 다시 설명합니다.

학습 완료 기준
처음 보는 화면에서 행동 하나를 재현하고, “이 요청이 API 후보인 이유”를 시각·메서드·URL·Payload·Response·Initiator 중 네 가지 이상으로 설명할 수 있습니다.

2. API 찾기는 숨은 주소 찾기가 아닙니다

API(Application Programming Interface)는 두 프로그램이 정해진 규칙으로 기능과 데이터를 주고받는 접점입니다. 웹 화면은 사용자의 클릭을 JavaScript 동작으로 바꾸고, 필요한 경우 서버 API를 호출해 데이터를 받거나 상태를 바꿉니다.

그러나 Network에 보이는 모든 요청이 우리가 찾는 **업무 API**는 아닙니다.

그림 2. Network에는 화면을 구성하는 파일과 데이터를 주고받는 요청이 함께 나타납니다.

Doc

HTML 문서

화면의 뼈대

JS

JavaScript

화면 동작 코드

Img

이미지

사진과 아이콘

Font

글꼴

텍스트 표시

Fetch/XHR도 광고·분석일 수 있으므로

CSS

스타일

모양과 배치

Fetch/XHR

데이터 요청

업무 API의 유력 후보

URL·Payload·Response를 확인하다

Type	주로 받는 것	API 후보 판단
`Doc`	HTML 문서	첫 화면·페이지 이동의 문서 요청일 수 있음
`JS`	JavaScript 코드	화면 동작을 구현하는 파일이지 보통 업무 데이터 API는 아님
`CSS`	스타일 규칙	화면 모양을 위한 파일
`Img`	이미지	상품 이미지·아이콘·배너
`Font`	글꼴	문자 표시용 자원
`Fetch/XHR`	JSON·텍스트·파일 등	업무 API의 유력 후보이지만 분석·광고 요청일 수도 있음

Fetch와 XHR

- Fetch: 브라우저의 `fetch()` API로 시작한 네트워크 요청
- XHR: `XMLHttpRequest` 객체로 시작한 요청
- Chrome의 `Fetch/XHR` 유형 필터: 두 종류를 함께 좁혀 보는 필터

`Fetch/XHR`만 선택했다고 업무 API가 확정되는 것은 아닙니다. 사용자 행동과 무관한 상태 확인, 오류 수집, 광고, 분석 요청도 여기에 보일 수 있습니다.

BIG IDEA 01

API 후보는 이름이 아니라 화면 행동과 함께 변한 증거로 찾습니다.

30초 확인

`logo.svg`와 `products?query=keyboard`가 같은 시각에 보입니다. 어느 쪽을 먼저 API 후보로 확인할까요?

정답 보기

`products?query=keyboard`를 먼저 확인합니다. 다만 이름만으로 확정하지 않고 Type, 메서드, 질의, 응답, 행동 시각을 함께 봅니다.

3. 좋은 API 찾기는 작은 통제 실험입니다

여러 버튼을 연속으로 누르면 어떤 요청이 어느 행동에서 생겼는지 분리하기 어렵습니다. 과학 실험처럼 바꾸는 조건을 하나로 제한합니다.

그림 3. 기록을 비우고 행동 하나만 실행하면 전과 후의 차이가 API 후보가 됩니다.


전
기록 지우기
필요하면 Preserve log
캐시 조건 설정
행동
버튼 한
상품 조회처럼 의미
좋은 실험: 행동 하나 ·
나쁜 실험: 여러 버튼 클릭 · 자동

짧은 시간 · 재현 가능

새로고침 섞임 · 기록 시각 없음

전 · 기준 상태 만들기

1. Network를 엽니다.
2. 기록 버튼이 켜져 있는지 확인합니다.
3. Clear로 이전 요청을 지웁니다.
4. 필요할 때만 Preserve log와 Disable cache를 설정합니다.

행동 · 의미가 분명한 동작 하나

- **상품 조회** 한 번
- **저장** 한 번
- 목록의 **다음 페이지** 한 번
- 검색어 입력 후 **Enter** 한 번

행동 직전 시각과 화면에 보인 입력을 기억합니다. 자동 완성처럼 입력 중 여러 번 호출되는 기능은 먼저 타이핑을 멈춘 뒤 마지막 요청 묶음을 관찰합니다.

후 · 새 요청과 화면 결과 연결

- 새로 생긴 요청 개수
- 행동 직후의 시작 시각
- 성공·실패한 화면 결과
- 요청의 Method·URL·Status·Type
- 입력과 닮은 Query String·Form Data·Request Payload
- 화면 결과와 닮은 Preview·Response

실험 품질	예시
좋음	Clear → 상품 조회 한 번 → 새 요청 4개 비교
좋음	같은 행동을 두 번 재현해 공통으로 생긴 요청 표시
나쁨	검색·필터·페이지 이동을 연속 실행한 뒤 추측
나쁨	Network를 늦게 열어 요청이 빠진 상태로 결론

재현 규칙: 행동 하나, 짧은 관찰 구간, 기록된 시각, 같은 조건에서 한 번 더 확인.

4. Network 작업 공간을 준비합니다

4.1. 개발자 도구 열기

운영체제	개발자 도구
Windows·Linux	`F12` 또는 `Ctrl` + `Shift` + `I`
macOS	`Option` + `Command` + `I`
공통	Chrome 메뉴 → 도구 더보기 → 개발자 도구

Network 탭을 선택합니다. 개발자 도구를 연 뒤 화면을 새로 고치거나 행동을 다시 실행해야 요청이 기록됩니다.

4.2. 자주 쓰는 네 가지 조작

조작	의미	언제 쓰는가
Record	요청 기록 시작·중지	빨간 기록 아이콘이 켜졌는지 확인
Clear	현재 요청 목록 삭제	행동 전 기준 상태 만들기
Preserve log	페이지 이동 뒤에도 이전 요청 보존	로그인·리다이렉션·페이지 전환 추적
Disable cache	개발자 도구가 열린 동안 캐시 우회	새 요청이 실제로 나가는지 비교할 때

Disable cache는 시험 결과에 영향을 줍니다. 실제 사용자의 캐시 조건을 재현해야 할 때는 끄고, 설정 여부를 분석 기록에 남깁니다.

> **운영 서비스 주의:** 저장·삭제·결제·발송 버튼을 실습처럼 반복하지 않습니다. 읽기 전용 시험 화면이나 별도 시험 계정에서 먼저 연습합니다.

5. 다섯 단계 필터 사다리를 오릅니다

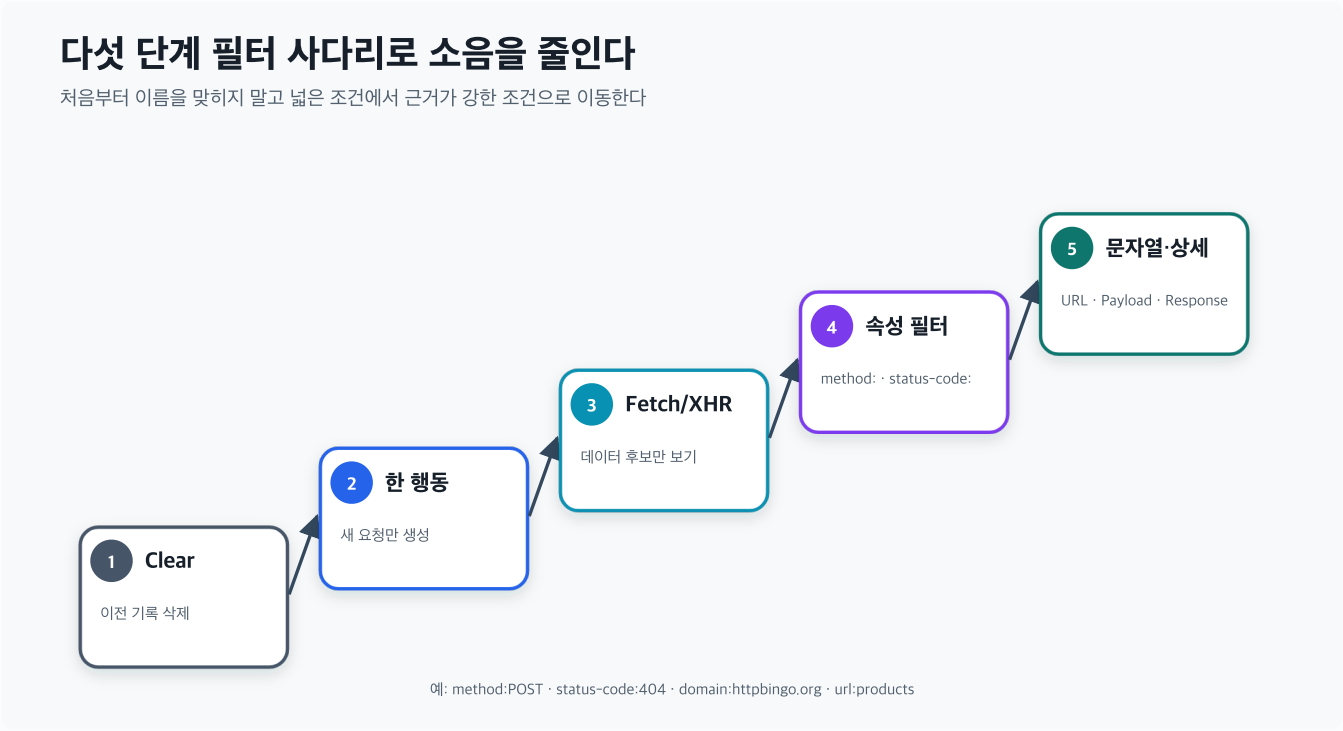
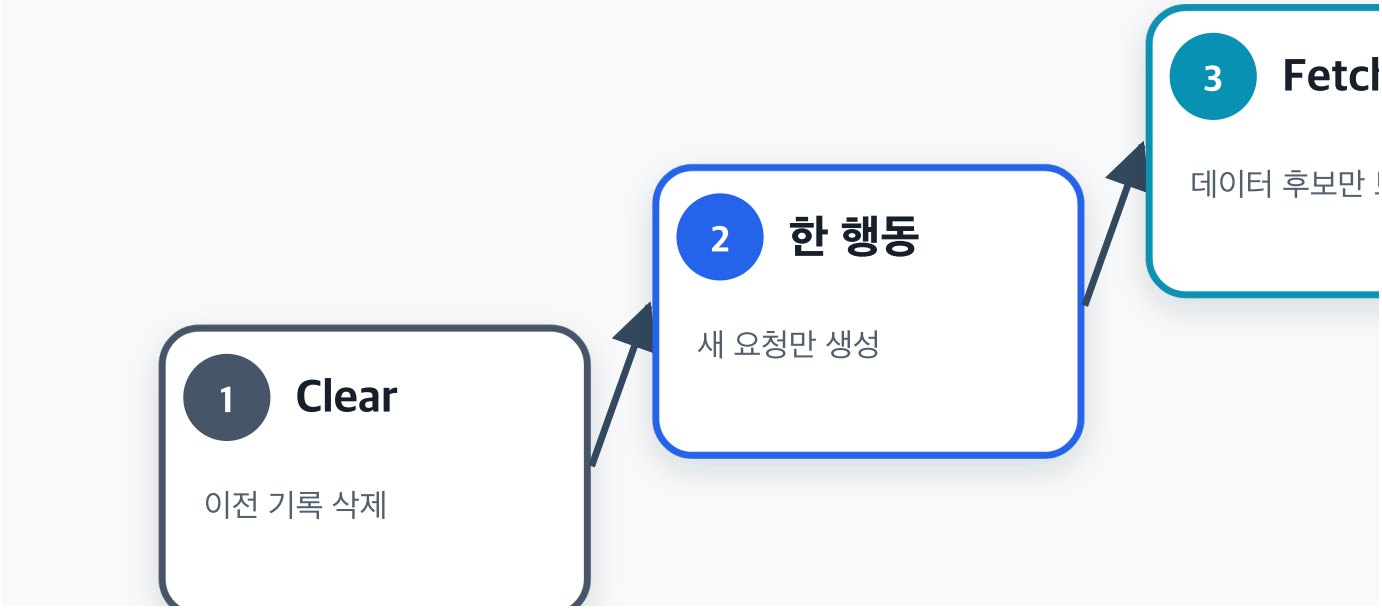
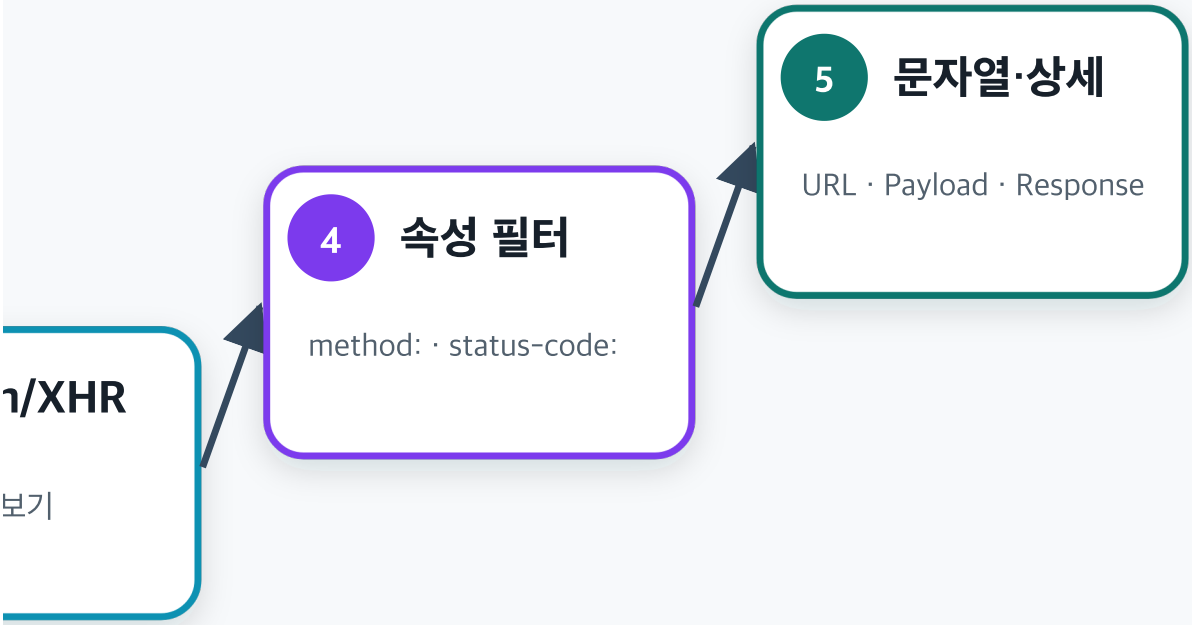


그림 4. 넓은 범위에서 시작해 Type, 속성, 상세 근거 순서로 후보를 좁힙니다.


1
Clear
이전 기록 삭제
2
한 행동
새 요청만 생성
3
Fetcl
데이터 후보만 !


5
문자열·상세
URL · Payload · Response
4
속성 필터
method: · status-code:
ı/XHR
보기

1단계 · Clear

이전 요청을 지워 지금 실행할 행동과 무관한 줄을 제거합니다.

2단계 · 행동 하나 실행

버튼을 한 번 누르고 요청이 생긴 시각을 기억합니다.

3단계 · Fetch/XHR

데이터 요청 후보를 우선 봅니다. 후보가 없으면 `All`로 돌아가 `Doc`, `Other`, WebSocket 등 다른 유형도 확인합니다.

4단계 · 문자열과 속성 필터

Chrome Network 필터에는 문자열과 속성을 함께 사용할 수 있습니다.

필터 예시	보는 범위
`products`	URL에 `products`가 포함된 요청
`domain:httpbingo.org`	지정 도메인 요청
`method:POST`	POST 요청
`status-code:404`	상태 코드 404 요청
`url:filters`	URL에 `filters`가 포함된 요청

여러 속성을 공백으로 함께 입력하면 조건을 더 좁힐 수 있습니다. Chrome 버전에 따라 화면 배치와 지원 필터가 달라질 수 있으므로 자동 완성 목록도 확인합니다.

5단계 · 요청 상세 확인

목록의 한 요청을 선택하고 Headers, Payload, Preview, Response, Initiator, Timing을 읽습니다.

BIG IDEA 02

필터는 정답을 주는 기능이 아니라 비교할 후보 수를 줄이는 기능입니다.

요청 목록에서 먼저 볼 열

열	질문
Name	어느 자원 또는 경로처럼 보이는가
Status	HTTP 응답을 받았는가, 실패했는가
Type	Doc·Fetch·XHR·Img 중 무엇인가
Initiator	어떤 문서·스크립트·요청이 시작했는가
Size	네트워크 전송과 자원 크기는 어느 정도인가
Time	전체 처리 시간이 얼마나 걸렸는가
Waterfall	언제 시작해 어느 요청과 겹쳤는가

표 머리글을 마우스 오른쪽 버튼으로 선택하면 Method, Protocol, Domain 같은 열을 추가할 수 있습니다.

6. API 후보를 여섯 가지 증거로 검증합니다

그림 5. 화면 행동과 요청의 여섯 근거가 서로 맞을수록 업무 API 후보라는 판단이 강해집니다.

WHEN

행동 직후인가

15:27:54

METHOD

무슨 의도인가

GET

PAYLOAD

화면 입력이 있는가

query=keyboard

RESPONSE

화면 결과와 맞는가

200 · items[]

여성 증거가 화면 행동과 일치

URL

어느 자원인가

`/products`

INITIATOR

누가 호출했는가

`app.js:142`

하면 “API 후보”로 기록한다

증거	확인 위치	좋은 질문
When	Waterfall·시작 시각	버튼을 누른 직후 생겼는가
Method	Headers의 General	조회·등록·변경 의도와 맞는가
URL	General의 Request URL	자원·행동·환경을 나타내는가
Payload	Payload	화면 입력·선택값이 들어 있는가
Response	Preview·Response	화면 결과와 닮은 데이터가 있는가
Initiator	Initiator	화면 동작 코드가 이 요청을 시작했는가

6.1. Headers

Headers는 요청과 응답의 기본 사실을 봅니다.

- General: Request URL, Request Method, Status Code
- Request Headers: 보낸 형식·인증·출처 등의 조건
- Response Headers: 받은 형식·캐시·CORS·추적 조건
- Query String Parameters: URL 질의를 해석한 값

6.2. Payload

화면에서 입력하거나 선택한 값이 서버로 어떻게 전달됐는지 봅니다.

- Query String Parameters
- Form Data
- Request Payload

비밀번호·토큰·개인정보가 보이면 캡처 전에 가립니다. Network에 보인다고 외부 공유가 허용된 정보는 아닙니다.

6.3. Preview와 Response

- Preview: JSON 등을 접어서 읽기 편하게 표현
- Response: 받은 원문에 가까운 내용

화면의 상품명·개수·상태가 응답 데이터와 맞는지 비교합니다. 응답이 JSON이 아니라 HTML이면 로그인 화면·오류 페이지·프록시 안내문을 받은 것은 아닌지 확인합니다.

6.4. Timing

요청의 대기·연결·서버 응답 대기·다운로드 구간을 봅니다. Timing만으로 서버 내부 원인을 확정할 수는 없지만, 지연이 연결 전인지 응답 대기인지 구분하는 단서가 됩니다.

후보 신뢰도 점검

일치한 증거	기록 방식
1~2개	추측 단계, 추가 비교 필요
3~4개	유력 후보, 한 번 더 재현
5~6개	강한 후보, 명세·코드·담당자 확인으로 확정

이 점수는 표준이 아니라 학습용 판단 보조 도구입니다. API의 공식 이름과 계약은 API 명세·코드·담당 조직에서 최종 확인합니다.

7. GET과 POST를 화면 행동에 연결합니다

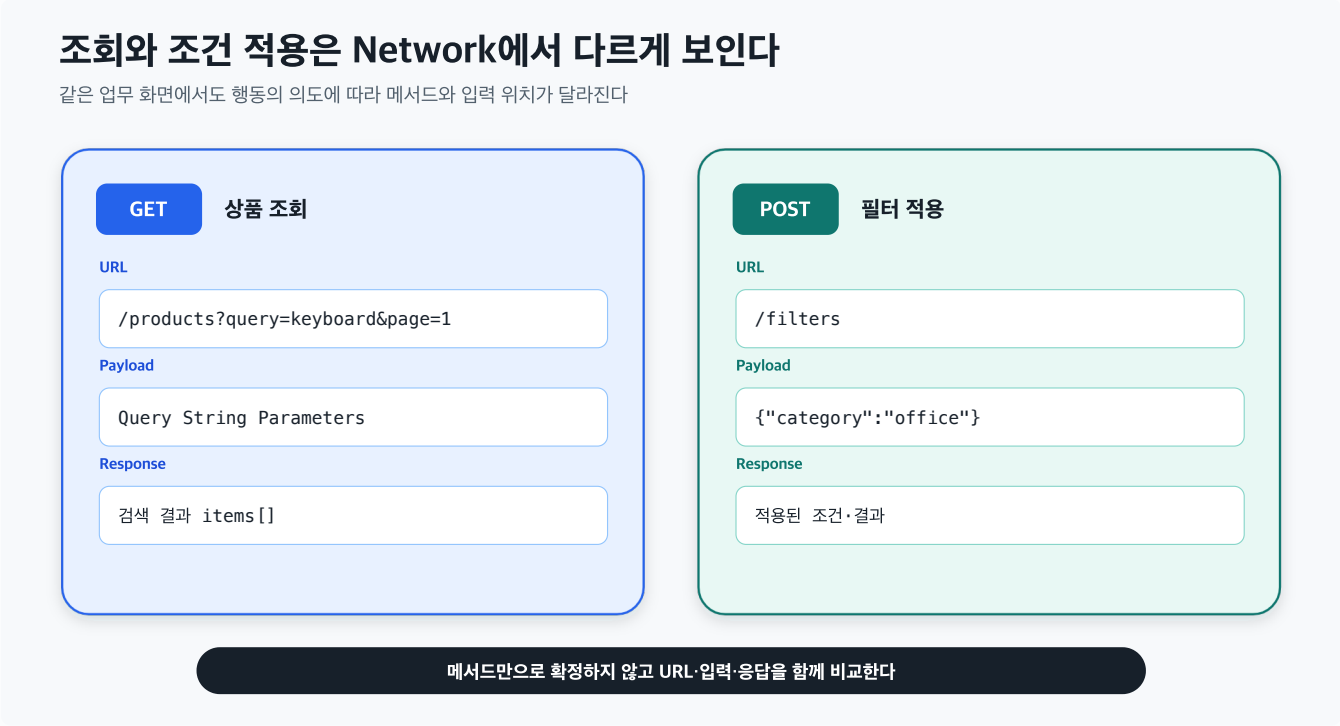


그림 6. 조회와 조건 적용은 메서드와 입력 위치가 다르지만 모두 화면 행동과 함께 읽어야 합니다.

GET

상품 조회

URL

```
/products?query=keyboard&page=1
```

Payload

```
Query String Parameters
```

Response

```
검색 결과 items[]
```

POST

필터 적용

URL

```
/filters
```

Payload

```
{"category":"office"}
```

Response

적용된 조건·결과

상품 조회 예시

```
GET https://httpbingo.org/anything/gibalja/products?query=keyboard&page=1
```

근거	관찰 값
화면 행동	상품 조회
Method	`GET`
경로	`/anything/gibalja/products`
Query String	`query=keyboard` , `page=1`
Status	`200`
Response	서버가 받은 method·url·args를 JSON으로 반환

필터 적용 예시

```
POST https://httpbingo.org/anything/gibalja/filters
```

```
{
  "category": "office",
  "inStock": true
}
```

근거	관찰 값
화면 행동	필터 적용
Method	`POST`
경로	`/anything/gibalja/filters`
Request Payload	`category` , `inStock`
Status	`200`

근거	관찰 값
Response	서버가 받은 JSON과 요청 정보를 반환

비교 문장: 상품 조회는 질의를 URL에 담은 GET 후보이고, 필터 적용은 JSON을 Request Payload에 담은 POST 후보입니다.

`fetch()`의 성공과 HTTP 성공은 다릅니다

브라우저의 `fetch()` Promise는 서버가 `404` 같은 HTTP 오류 상태를 보내도 일반적으로 응답 객체를 받을 수 있습니다. 따라서 화면 코드는 `response.ok` 또는 `response.status`를 확인해야 합니다.

```
const response = await fetch(url);
if (!response.ok) {
  // 404, 500 등 HTTP 오류 상태 처리
}
```

Network에서 `404` 응답이 보인다면 “통신이 전혀 안 됐다”가 아니라 HTTP 응답을 받았다는 뜻입니다. DNS·연결·CORS 실패처럼 상태 코드를 받지 못한 경우와 구분합니다.

8. OPTIONS와 실제 요청을 짝으로 봅니다

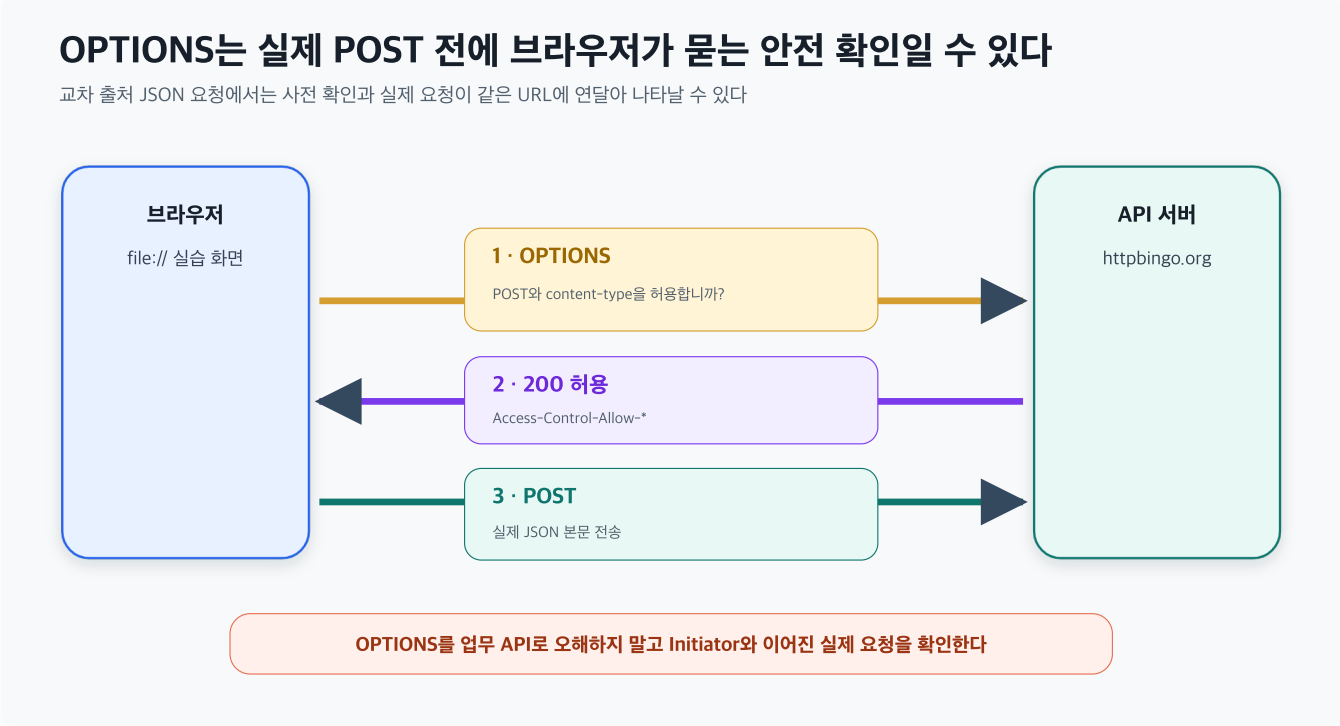
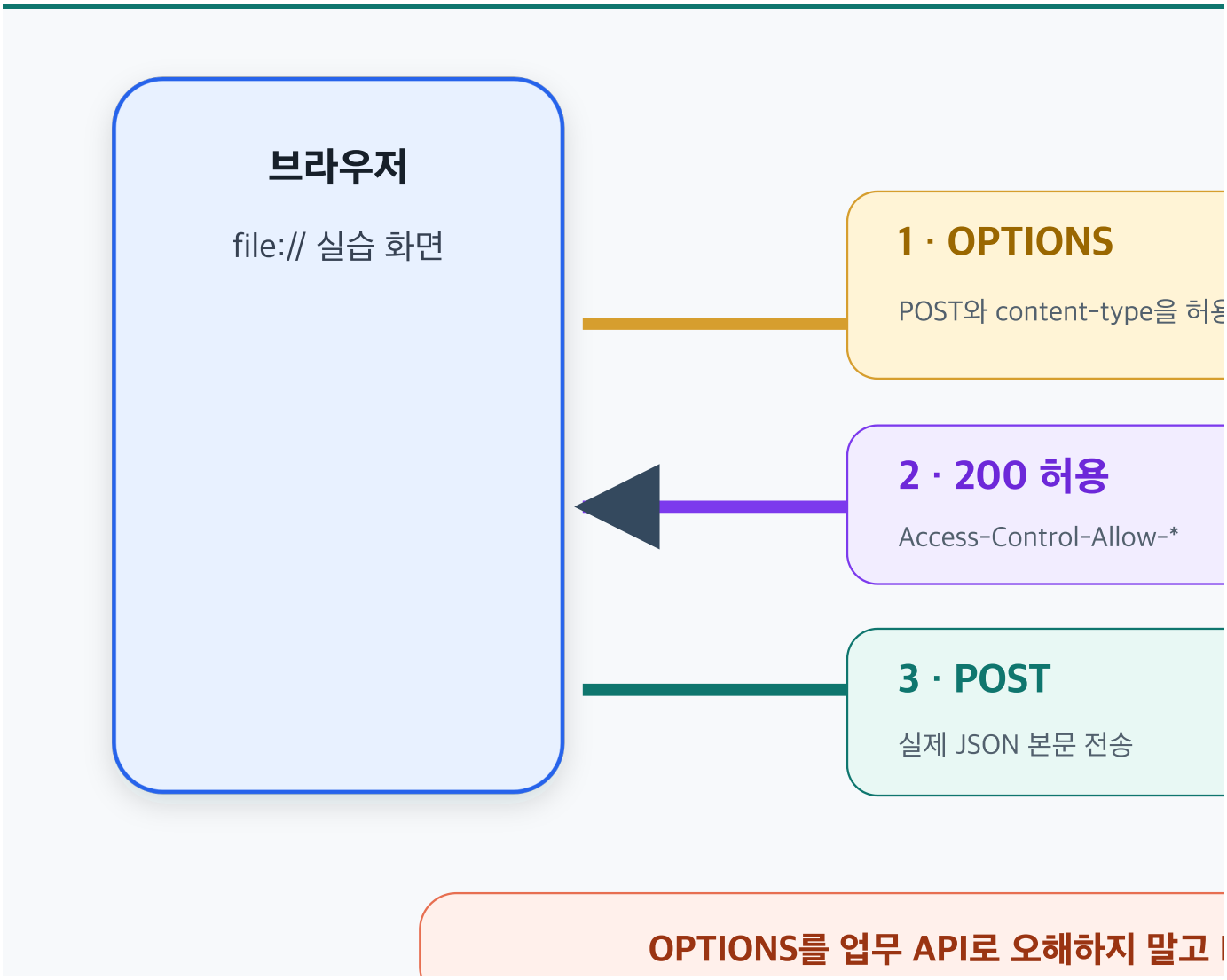
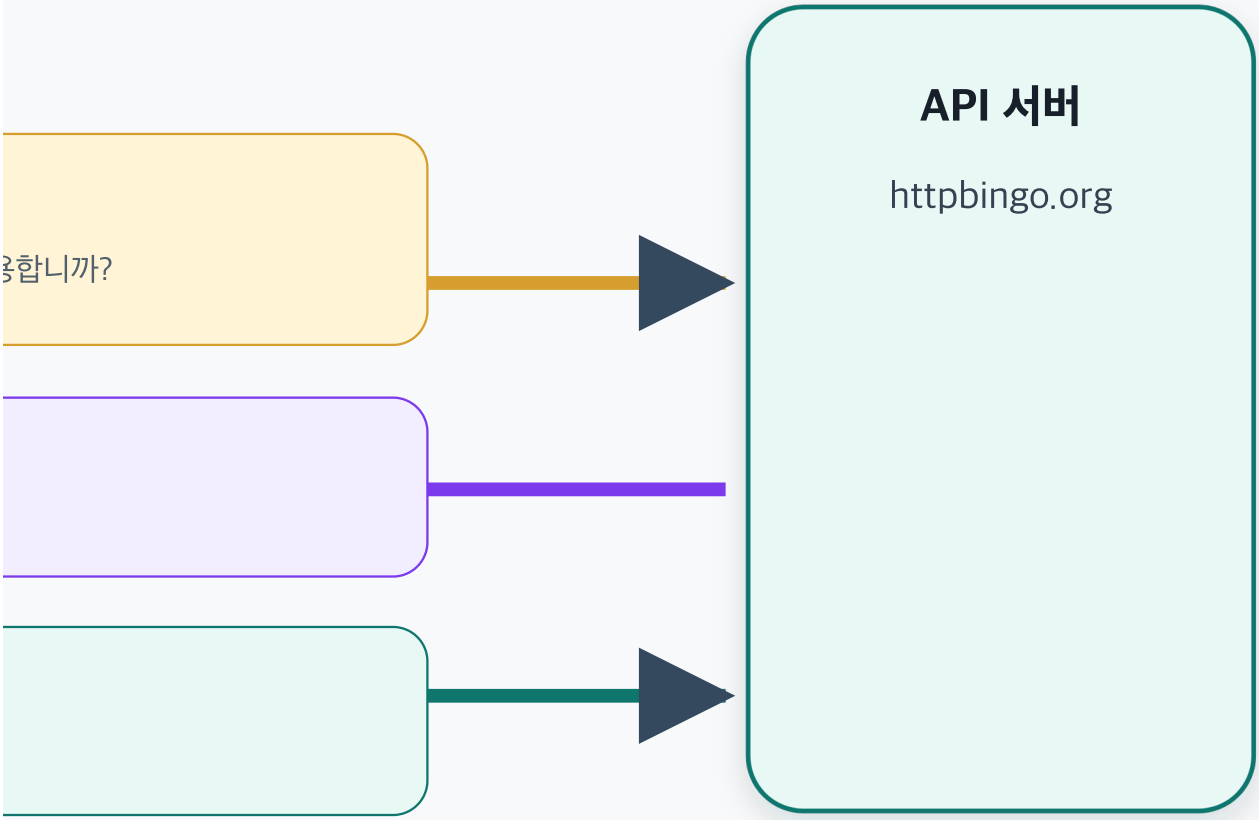


그림 7. 브라우저는 교차 출처의 일부 요청을 보내기 전에 OPTIONS로 허용 조건을 확인합니다.


브라우저
file:// 실습 화면
1 · OPTIONS
POST와 content-type을 허용
2 · 200 허용
Access-Control-Allow-*
3 · POST
실제 JSON 본문 전송
OPTIONS를 업무 API로 오해하지 말고



Initiator와 이어진 실제 요청을 확인한다

실습 HTML 파일과 `https://httpbingo.org`는 출처(Origin)가 다릅니다. 브라우저는 JSON POST를 보내기 전에 다음을 자동으로 수행할 수 있습니다.

1. `OPTIONS /anything/gibalja/filters`
2. 서버가 허용 메서드·헤더·출처를 응답
3. 조건이 맞으면 실제 `POST /anything/gibalja/filters`

두 요청을 구분하는 표

항목	OPTIONS	POST
목적	실제 요청을 보내도 되는지 사전 확인	업무 입력을 실제로 전송
Initiator	`preflight`로 표시될 수 있음	`script`·호출 코드
Request Headers	`Access-Control-Request-Method`, `Access-Control-Request-Headers`	`Content-Type`, 실제 인증·업무 헤더
Payload	보통 실제 업무 JSON 없음	`category`, `inStock` 등 실제 입력
결론	업무 API의 앞단 확인 요청	화면 행동과 직접 연결할 유력 후보

CORS 실패를 만났을 때

- Console의 CORS 오류 문구 확인
- OPTIONS의 상태와 응답 헤더 확인
- 실제 요청이 전송됐는지 확인
- 요청 Origin과 서버 허용 Origin 비교
- 허용 Method와 Header 비교
- 브라우저 보안을 임의로 끄지 말고 서버·게이트웨이 정책 담당자에게 증거 전달

> **오해 주의:** OPTIONS가 실패하면 실제 POST가 전송되지 않을 수 있습니다. 이때 POST의 서버 업무 로그만 찾으면 원인을 놓칠 수 있습니다.

30초 확인

같은 URL에 `OPTIONS 200`과 `POST 200`이 연달아 보이고, JSON은 POST Payload에만 있습니다. 화면의 필터 적용과 직접 연결할 요청은 무엇일까요?

정답 보기

실제 업무 입력을 담은 POST입니다. OPTIONS는 브라우저가 먼저 보낸 CORS 사전 확인 요청으로 함께 기록합니다.

9. Initiator로 호출한 주체를 추적합니다

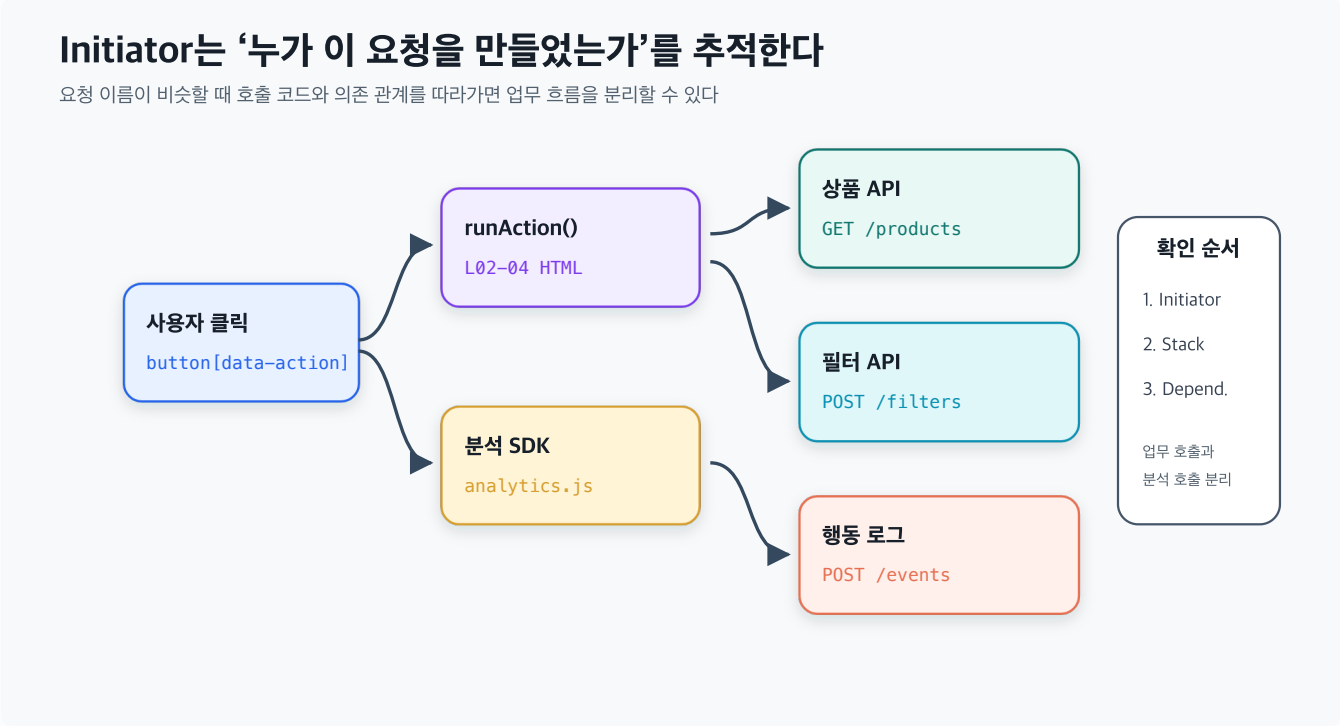
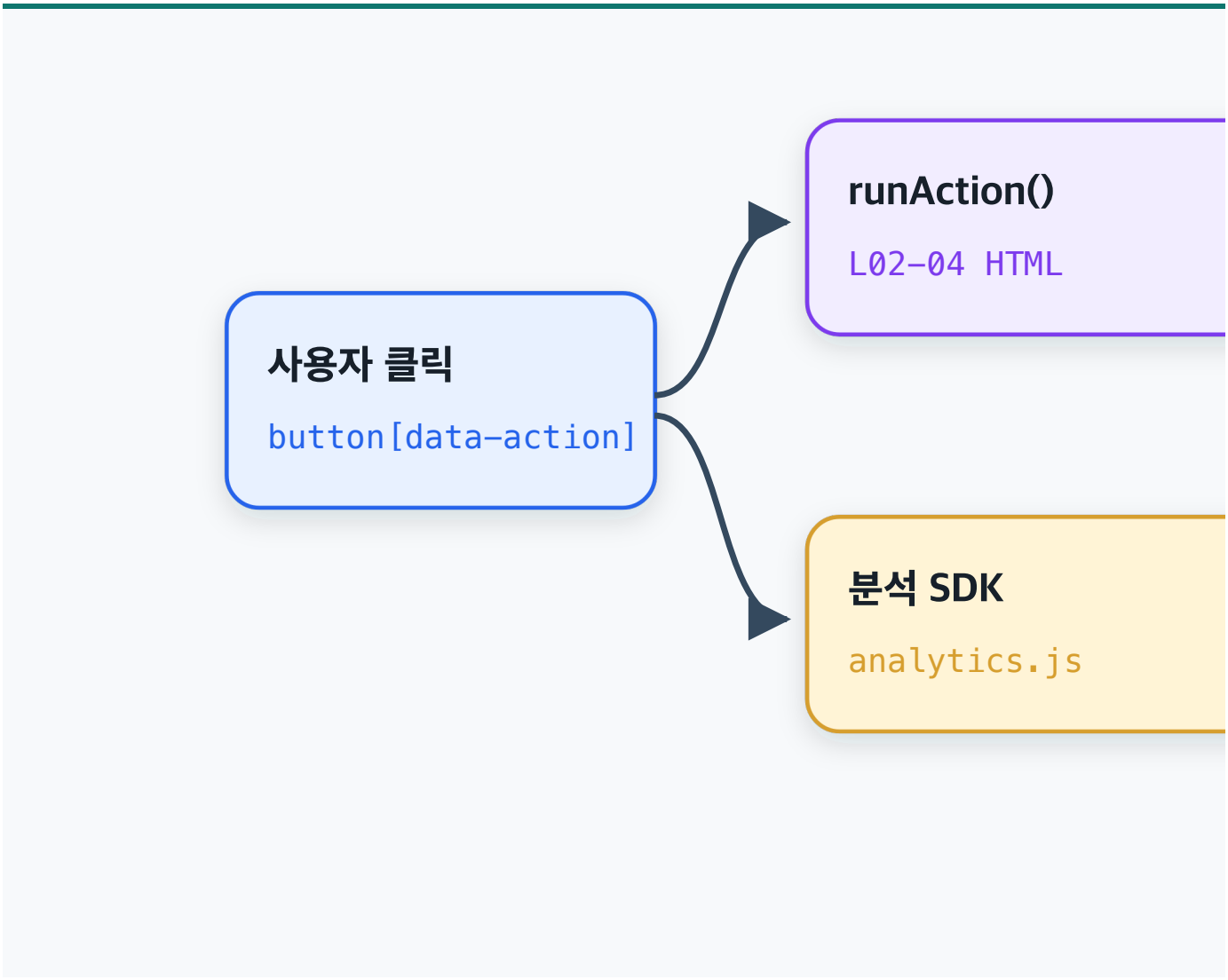
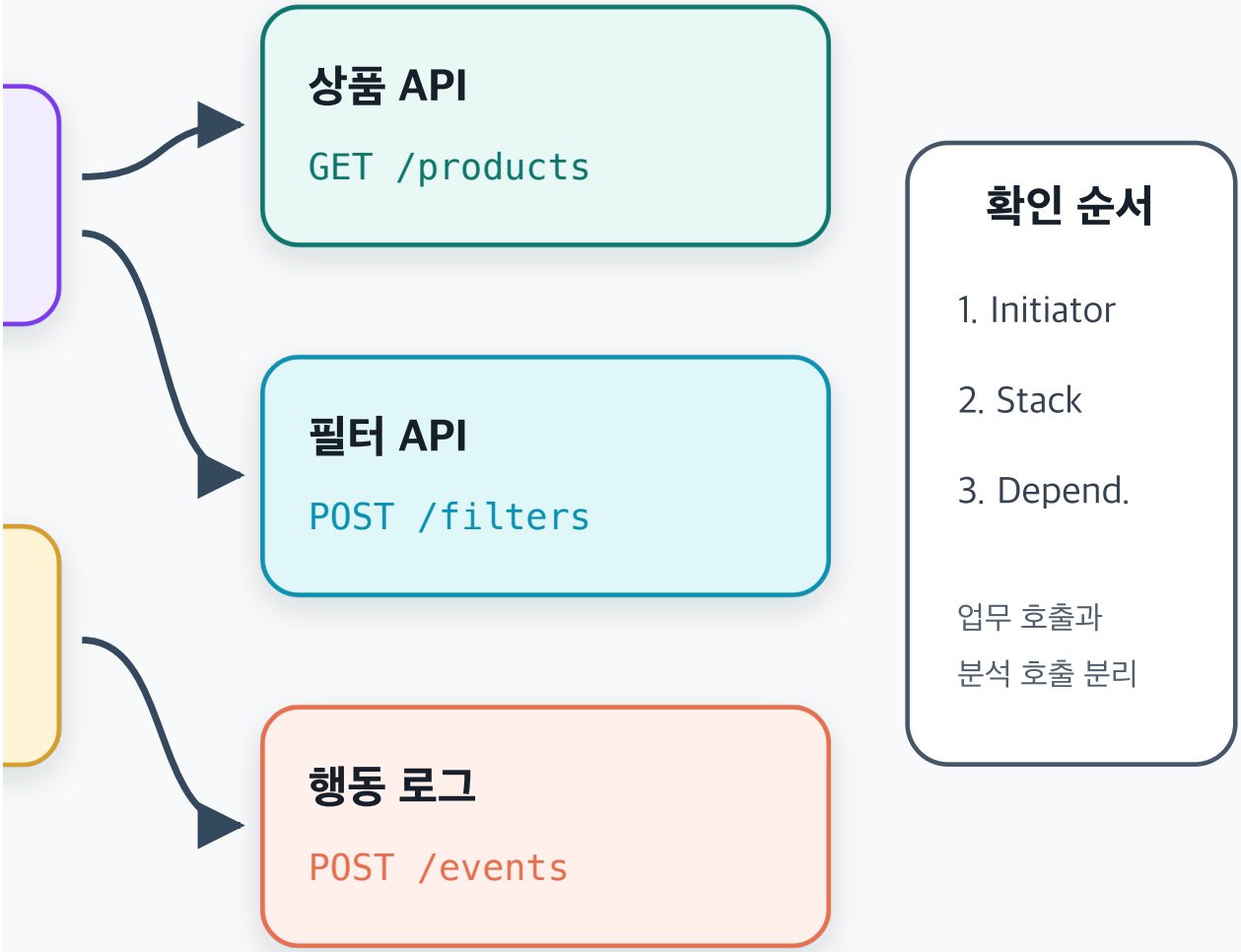


그림 8. 같은 클릭에서 업무 API와 분석 요청이 함께 생기면 Initiator가 호출 흐름을 분리하는 단서가 됩니다.


사용자 클릭
button[data-action]
runAction()
L02-04 HTML
분석 SDK
analytics.js


상품 API
GET /products
필터 API
POST /filters
행동 로그
POST /events
확인 순서
1. Initiator
2. Stack
3. Depend.
업무 호출과
분석 호출 분리

Initiator에서 볼 것

- Parser: HTML을 읽다가 발견한 자원
- Script: JavaScript가 시작한 요청
- Preflight: CORS 사전 확인
- Redirect: 앞 응답의 이동 지시에 이어진 요청
- 호출 스택: 어느 함수와 코드 위치에서 시작했는지

요청 목록의 Initiator 열이나 요청 상세의 Initiator 탭에서 링크를 선택하면 Sources의 관련 코드로 이동할 수 있습니다. 난독화·번들링된 운영 코드는 이름이 읽기 어려울 수 있으므로 URL·Payload·Response 증거와 함께 봅니다.

요청 의존 관계

Chrome은 Initiator와 dependency 관계를 이용해 요청이 무엇을 시작했고 무엇에서 시작됐는지 보여 줄 수 있습니다. 다음 경우에 유용합니다.

- 로그인 요청 뒤 사용자 정보 요청이 이어짐
- 사전 확인 OPTIONS 뒤 실제 POST가 이어짐
- 설정 API 응답 뒤 여러 목록 API가 이어짐
- 리다이렉션 뒤 새 문서 요청이 이어짐

업무 API와 분석 요청 구분

질문	업무 API 쪽 단서	분석 요청 쪽 단서
URL	상품·주문·사용자 등 업무 자원	events·collect·analytics 등 추적 이름
Payload	화면 입력·업무 식별자	화면명·이벤트명·클릭 좌표
Response	화면에 표시할 데이터·업무 상태	빈 응답·수집 완료 표시
Initiator	화면 기능 코드	분석 SDK·태그 관리자

이름은 어디까지나 단서입니다. 실제 서비스에서는 분석 API도 중요한 시스템 구성요소이며, 이름이 다르게 설계될 수 있습니다.

10. Network 검색과 복사 기능을 사용합니다

10.1. 전체 요청 내용 검색

Network의 검색 기능은 여러 요청의 Headers, Payload, Response에서 문자열을 찾을 수 있습니다. 화면에 보인 고유한 상품 ID나 오류 코드가 어느 요청에 있는지 찾을 때 유용합니다.

검색할 값은 다음처럼 선택합니다.

- 짧고 흔한 `200`, `id` 대신 고유한 `product-481`
- 개인정보 대신 시험용 가상 식별자
- 화면 오류 코드·업무 상태·목록 항목명

10.2. 다시 실행과 복사

요청을 마우스 오른쪽 버튼으로 선택하면 버전과 요청 유형에 따라 다음 기능을 사용할 수 있습니다.

기능	용도	주의
Replay XHR	XHR 요청 재실행	등록·결제·삭제 요청은 중복 처리 위험
Copy URL	전체 URL 복사	질의에 토큰·개인정보가 없는지 확인
Copy as cURL	명령줄에서 재현할 형태로 복사	Authorization·Cookie가 포함될 수 있음
Copy as fetch	브라우저 fetch 코드로 복사	자격 정보와 실행 환경 확인
Copy response	응답 본문 복사	개인정보·내부 데이터 확인

재실행 주의: POST·PATCH·DELETE는 실제 업무 상태를 바꿀 수 있습니다. 화면에서 한 번, Network에서 한 번 재실행하면 두 번 처리될 수 있습니다.

11. 증거는 안전하게 공유합니다

그림 9. 재현에 필요한 사실은 남기고 인증 자격과 개인정보는 제거합니다.

반드시 가릴 것

Authorization · Cookie · Set-Cookie · API 키

이메일 · 전화번호 · 내부 주소 · 업무 데이터

3 민감정보 가림

토큰 · 쿠키 · 개인정보

4 전달 전 재검토

환경 · 시각 · 재현 단계

함께 남길 것

발생 시각 · 환경 · 화면 행동 · 기대/실제 결과

메서드 · URL 구조 · 상태 · 요청 ID · Timing

공유할 최소 증거

1. 발생 시각과 표준시간대
2. 시험·개발·운영 환경
3. 화면 행동과 입력 조건
4. 기대 결과와 실제 화면 결과
5. Method와 URL 구조
6. Status와 중요한 Response 요약
7. 요청·추적 ID와 Timing

가리거나 제거할 정보

- `Authorization` 헤더와 접근 토큰
- `Cookie`, `Set-Cookie`, 세션 식별자
- API 키·서명·일회용 코드
- 이름·이메일·전화번호·주소 등 개인정보
- 비공개 내부 URL·IP·시스템 이름
- 주문·계약·인사·재무 등 업무 민감 데이터

HAR를 다룰 때

HAR(HTTP Archive)는 여러 네트워크 요청을 구조화해 저장하는 파일입니다. Chrome의 **sanitized HAR** 내보내기는 기본적으로 Cookie, Set-Cookie, Authorization 같은 일부 민감 필드를 제외하도록 설계되어 있습니다. 그러나 URL 질의·본문·응답의 업무 데이터까지 모두 안전해진다고 가정하면 안 됩니다.

내보낸 파일을 다시 열어 다음을 검색합니다.

```
Authorization
Cookie
Set-Cookie
token
email
phone
```

BIG IDEA 03

증거의 품질은 많이 담는 데서가 아니라 재현에 필요한 사실만 안전하게 남기는 데서 나옵니다.

12. 실습 · 세 행동에서 API 후보 찾기

연결 파일: L02-04 API 관찰 실습

실습 화면: L02-04 API 관찰 화면

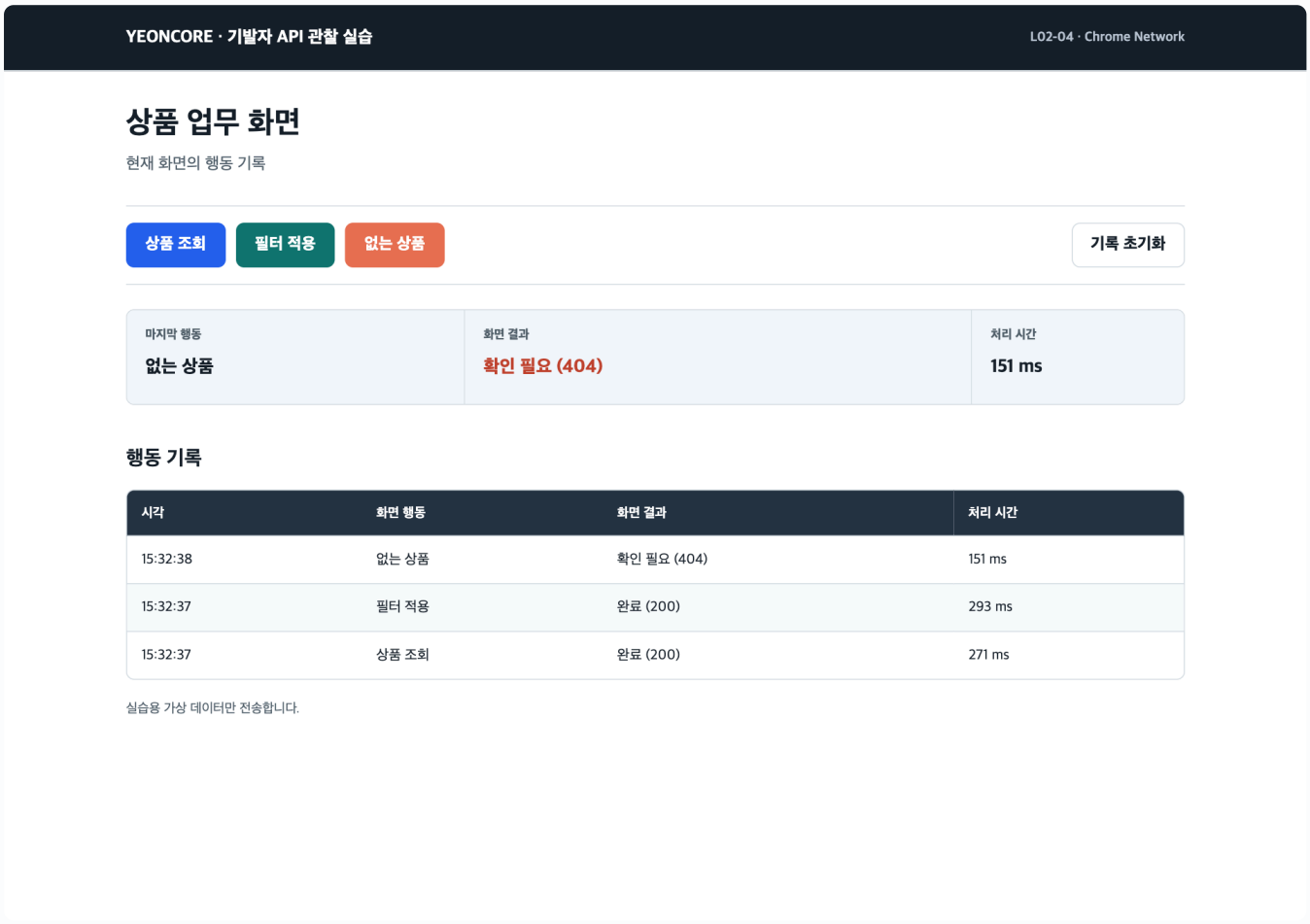


그림 10. 실습 화면의 버튼은 GET 200, OPTIONS와 POST 200, GET 404를 재현 가능하게 만듭니다.

준비

1. Chrome에서 실습 HTML을 엽니다.
2. 개발자 도구의 Network를 엽니다.
3. Clear를 누릅니다.
4. `Fetch/XHR` 필터를 선택합니다.

실습 A · 상품 조회

1. **상품 조회**를 한 번 누릅니다.

2. `products`가 포함된 요청을 선택합니다.
3. Method, URL, Status, Query String, Response를 기록합니다.

실습 B · 필터 적용

1. Clear를 누릅니다.
2. **필터 적용**을 한 번 누릅니다.
3. 같은 URL의 OPTIONS와 POST를 찾습니다.
4. POST의 Request Payload에서 `category`와 `inStock`을 확인합니다.
5. Initiator에서 preflight와 script를 구분합니다.

실습 C · 없는 상품

1. Clear를 누릅니다.
2. **없는 상품**을 한 번 누릅니다.
3. `status/404` 요청을 선택합니다.
4. 화면의 `확인 필요 (404)`와 Network의 Status를 연결합니다.

예상 관찰값

화면 행동	Method	URL 핵심	입력 위치	예상 상태
상품 조회	GET	`/products`	Query String	200
필터 적용 사전 확인	OPTIONS	`/filters`	사전 확인 헤더	200
필터 적용 실제 요청	POST	`/filters`	Request Payload	200
없는 상품	GET	`/status/404`	없음	404

외부 시험 서비스 상태와 네트워크 정책에 따라 연결 자체가 실패할 수 있습니다. 이 경우 실패 시각·Console 문구·Network 상태를 기록하고 API 구조 학습은 예시 값으로 계속합니다.

완료 산출물

- 화면 행동·API 지도 3행
- OPTIONS·POST 비교표 1개
- 민감정보 없는 요청 증거 1개
- 개발자에게 전달할 한 문장 1개

13. 화면 행동·API 분석 학습지

A. 행동 전·후 기록

항목	기록
발생 시각·표준시간대	
환경	
행동 전 화면 상태	
실행한 행동 하나	
입력·선택 조건	
기대 결과	
실제 화면 결과	
새로 생긴 요청 수	

B. API 후보 여섯 증거

증거	관찰값	화면 행동과 일치?
When		예·아니요·불명
Method		예·아니요·불명
URL		예·아니요·불명
Payload		예·아니요·불명
Response		예·아니요·불명
Initiator		예·아니요·불명

C. 화면 행동·API 지도

화면·행동	API 후보	Method	입력	결과	확신도·근거

D. 요청 쌍 그리기

```
[화면 행동]
    |
    ├──> [사전 요청·앞 요청] _____________________________
    |          상태 ______   역할 _____________________________
    |
    └──> [실제 업무 요청] _____________________________
               상태 ______   입력 _____________________________
```

E. 전달용 한 문장

_____ 시각 _____ 환경에서 화면의 _____ 행동을 한 번 실행했을 때 `_____` 요청이 발생했습니다. `_____` 메서드와 _____ 입력, _____ 응답이 화면 결과와 일치해 API 후보로 판단했으며, 상태는 _____입니다.

14. 자주 막히는 지점

증상	먼저 확인	다음 행동
요청 목록이 비어 있음	Record 상태·개발자 도구를 연 시점	Network를 연 채 행동 재실행
요청이 너무 많음	Clear·행동 개수	한 행동만 실행하고 Fetch/XHR 선택
Fetch/XHR에 후보가 없음	All의 Doc·Other·WS	전체 Type에서 행동 시각 기준 비교
Payload 탭이 없음	GET 질의·본문 없는 요청 여부	Headers의 Query String 확인
POST가 보이지 않음	OPTIONS 실패·Console CORS 오류	사전 확인 응답 헤더와 서버 정책 확인
Status가 `(failed)`	DNS·연결·TLS·CORS·취소	Console과 오류 문구·Timing 확인

증상	먼저 확인	다음 행동
응답이 HTML임	로그인 만료·프록시 오류·리다이렉션	Response 내용과 최종 URL 확인
같은 요청이 반복됨	자동 완성·폴링·재시도	Initiator·시간 간격·Payload 비교
화면은 바뀌지만 요청 없음	로컬 계산·캐시·미리 받은 데이터	Initiator와 Application 상태 확인
Copy as cURL 공유가 걱정됨	토큰·쿠키·본문	가리고 최소 재현 정보만 전달

API가 보이지 않아도 실패한 분석은 아닙니다

화면 행동이 다음 방식으로 처리되면 새 API 요청이 없을 수 있습니다.

- 이미 받은 데이터를 브라우저에서 필터링
- Service Worker나 캐시에서 응답
- WebSocket 메시지로 통신
- 클릭이 네트워크와 무관한 UI 상태만 변경
- 요청이 지연·묶음 처리됨

“새 Fetch/XHR가 없다”는 관찰도 유효한 결과입니다. 범위를 넓혀 WS·Other·Application·Sources를 다음 분석 대상으로 정합니다.

15. 셀프 테스트

문제 1 · 시작점

Network에서 API를 찾을 때 가장 먼저 해야 할 행동은 무엇입니까?

A. 가장 긴 URL 선택
B. 기록을 지우고 화면 행동 하나 실행
C. 상태 200 요청 모두 복사
D. 이미지 요청 숨기기만 함

문제 2 · Type

`Fetch/XHR` 필터의 올바른 의미를 쓰세요.

문제 3 · 증거

API 후보를 검증하는 여섯 증거를 쓰세요.

문제 4 · Payload

화면에서 `category=office`를 선택했습니다. 후보 요청의 어느 탭에서 이 입력을 우선 확인합니까?

문제 5 · 응답

화면에 상품 10개가 보입니다. Response에서 무엇을 비교해야 합니까?

문제 6 · OPTIONS

같은 URL에 OPTIONS와 POST가 보입니다. 실제 업무 JSON은 POST에 있습니다. OPTIONS의 역할은 무엇입니까?

문제 7 · 404

`fetch()`로 호출한 요청이 Network에서 404 응답을 받았습니다. "네트워크 연결 자체가 실패했다"고 말해도 됩니까?

문제 8 · Initiator

같은 클릭에서 `/products`와 `/events`가 함께 발생했습니다. 어떤 근거로 업무 API와 분석 요청을 구분합니까?

문제 9 · 보안

Copy as cURL 또는 HAR를 공유하기 전에 반드시 제거하거나 확인할 정보 네 가지를 쓰세요.

문제 10 · 전달 문장

다음 관찰을 개발자에게 전달할 한 문장으로 바꾸세요.

```
시각: 2026-07-15 15:27 KST
환경: 로컬 실습 화면
행동: 필터 적용 한 번
요청: POST /anything/gibalja/filters
Payload: category=office, inStock=true
Status: 200
```

16. 정답과 해설

문제 1

정답: B. 이전 기록을 지우고 행동 하나만 실행해야 새 요청과 행동의 관계를 비교할 수 있습니다.

문제 2

Fetch API 또는 XMLHttpRequest로 시작된 요청을 좁혀 보는 유형 필터입니다. 업무 API를 자동으로 확정하는 필터는 아닙니다.

문제 3

When, Method, URL, Payload, Response, Initiator입니다.

문제 4

Payload 탭의 Query String Parameters, Form Data, Request Payload 중 해당 요청 형식에 맞는 구역을 확인합니다. GET 질의는 Headers에도 해석되어 보일 수 있습니다.

문제 5

응답 배열의 항목 수, 상품 식별자·이름·상태가 화면 표시와 맞는지 비교합니다. 화면이 가공한 값일 수 있으므로 완전히 같은 글자만 찾지 말고 구조와 대응 관계를 봅니다.

문제 6

교차 출처의 실제 요청을 보내기 전에 서버가 요청 Method·Header·Origin을 허용하는지 브라우저가 확인하는 CORS 사전 요청입니다.

문제 7

아닙니다. 404는 서버로부터 HTTP 응답을 받았다는 뜻입니다. DNS·연결·CORS 등으로 상태 코드 자체를 받지 못한 실패와 구분합니다.

문제 8

URL 이름뿐 아니라 Payload가 화면 입력과 맞는지, Response가 화면 결과와 맞는지, Initiator가 기능 코드인지 분석 SDK인지, 시각이 어떻게 연결되는지 비교합니다.

문제 9

Authorization·접근 토큰, Cookie·Set-Cookie, API 키, 개인정보·업무 민감 데이터, 내부 URL·IP 등을 확인하고 필요한 부분을 제거합니다. 네 가지 이상 쓰면 됩니다.

문제 10

예시 답안:

> 2026-07-15 15:27 KST 로컬 실습 화면에서 필터 적용을 한 번 실행했을 때 `POST /anything/gibalja/filters` 요청이 발생했고, `category=office`, `inStock=true` Payload가 화면 입력과 일치했으며 상태는 200이었습니다.

17. 한 장 요약

그림 11. 준비, 행동, 필터, 검증, 추적, 기록의 여섯 단계로 API 후보를 찾습니다.

1 **준비**

Network 열기

기록 지우기

2 **행동**

한 번만 실행

시각·결과 기억

4 **검증**

Method·URL

Payload·Response

5 **추적**

Initiator

OPTIONS 짝

경로: “이 API다”보다 “이 구간로

3 **필터**

Fetch/XHR

문자열·속성

6 **기록**

화면↔API 지도

민감정보 가림

단계	해야 할 일	남길 결과
1	Network 열기·Clear	깨끗한 기준 상태
2	화면 행동 한 번	행동 시각·입력·화면 결과
3	Fetch/XHR·문자열·속성 필터	비교할 후보 목록
4	Method·URL·Payload·Response 확인	행동과 일치하는 근거
5	Initiator·OPTIONS·의존 관계 확인	호출 흐름과 요청 쌍
6	민감정보를 가리고 기록	화면 행동·API 지도

최종 설명: “이 API다”가 아니라 “이 행동 직후 발생했고, 이 입력과 결과가 일치해 이 요청을 API 후보로 기록했다”고 말합니다.

18. 다음 학습과 출처

18.1. 다음 매뉴얼

M02-05 「방화벽·VPN·프록시·망 분리 이해하기」에서는 브라우저에서 찾은 API가 어떤 네트워크 경계와 접속 조건을 거쳐 연결되는지 배웁니다.

18.2. 함께 사용할 자료

- L02-04 API 찾기 실습
- L02-04 API 관찰 화면
- T02-04 화면 행동·API 지도
- 개발자 도구·API 탐색 용어집

18.3. 출처와 확인일

아래 공식 문서는 모두 2026-07-15에 확인했습니다.

- Google Chrome Developers: Inspect network activity, Network features reference, DevTools preferences
- MDN Web Docs: Fetch API, Using Fetch, CORS, OPTIONS