Day6

자료	Javascrtip
구분	Javascript

AJAX

AJAX란?

- Asynchronous JavaScript And XML (비동기식 JavaScript와 XML)
- 비동기 통신을 이용하면 화면 전체를 새로고침 하지 않아도 서버로 요청을 보내고, 데이터를 받아 화면의 일부분만 업데이트 가능
- 이러한 '비동기 통신 웹 개발 기술'을 AJAX라 함
- 비동기 웹 통신을 위한 라이브러리 중 하나가 Axios

AJAX 특징

- 페이지 전체를 reload(새로 고침)를 하지 않고서도 수행되는 "비동기성"
- 서버의 응답에 따라 전체 페이지가 아닌 일부분만을 업데이트 할 수 있음

1. 페이지 새로고침 없이 서버에 요청
2. 서버로부터 응답(데이터)을 받아 작업을 수행

비동기 적용하기 - 팔로우(follow)

axios 요청 준비

팔로우 기능을 비동기로 개선해보자.

먼저, `base.html` 을 다음과 같이 수정한다.

각각의 템플릿에서 script 코드를 작성하기 위한 block tag 영역 작성해 준다.

```
<!-- base.html -->
<body>
...
  {% block script %}
  {% endblock script %}
```

```
</body>
</html>
```

맨 아래에, `script` 를 넣을 수 있는 영역을 추가했다.

다음, `accounts/profile.html` 을 아래와 같이 수정한다.

axios CDN 작성할 것이다. profile.html 파일 안에

{% endblock %} 밑에 {% block script %} 을 넣고 CDN을 추가하자.

```
<!-- accounts/profile.html -->

{% block script %}
  <script src="https://cdn.jsdelivr.net/npm/axios/dist/axios.min.js"></script>

  <script>
  </script>
{% endblock script %}
```

- form 요소 선택을 위해 id 속성 지정 및 선택
  - 불필요해진 action과 method 속성은 삭제할 것이다. 요청은 axios로 대체되기 때문이다.

```
<!-- accounts/profile.html -->

//<form action="{% url 'accounts:follow' person.pk %}" method="POST">  삭제!
<form id="follow-form">
  {% csrf_token %}
  {% if request.user in person.followers.all %}
    <input type="submit" value="Unfollow" />
  {% else %}
    <input type="submit" value="Follow" />
  {% endif %}
</form>
```

```
<!-- accounts/profile.html -->

{% block script %}
  <script src="https://cdn.jsdelivr.net/npm/axios/dist/axios.min.js"></script>

  <script>
    const form = document.querySelector("#follow-form")
  </script>
{% endblock script %}
```

- form 요소에 이벤트 핸들러 작성 및 submit 이벤트 취소를 하겠다.

```
<!-- accounts/profile.html -->

<script>
  const form = document.querySelector("#follow-form")
  form.addEventListener('submit', function (event) {
    event.preventDefault() // form의 기본 기능 막음
  })
</script>
```

- axios 요청 준비

```
<script>
    const form = document.querySelector("#follow-form")

    form.addEventListener('submit', function (event) {
      event.preventDefault() // form의 기본 기능 막음
      axios({
        method: "post",
        url: "/accounts/${???}/follow/",
      })
    })

  </script>
```

url에 작성할 user pk 가져오기

- url에 작성할 user pk 가져오기 (HTML -> JavaScript)

```
<!-- accounts/profile.html -->

<form id="follow-form" data-user-id="{{ person.pk }}">
...
</form>
```

```
<!-- accounts/profile.html -->

<script>
  const form = document.querySelector('#follow-form’)

  form.addEventListener('submit', function (event) {
    event.preventDefault()
    const userId = event.target.dataset.userId
    ...
```

```
  })
  </script>
```

- url 작성 마치기

```
<!-- accounts/profile.html -->

<script>
    const form = document.querySelector("#follow-form")
    form.addEventListener('submit', function (event) {
      event.preventDefault() // form의 기본 기능 막음
      const userId = event.target.dataset.userId
      axios({
        method: "post",
        url: "/accounts/${userId}/follow/",

      })
    })

  </script>
```

[참고] data-* attributes

- 사용자 지정 데이터 특성을 만들어 임의의 데이터를 HTML과 DOM사이에서 교환 할 수 있는 방법이다. 사용 예시를 살펴보자.

```
<div data-my-id="my-data"></div>
<script>
  const myId = event.target.dataset.myId
</script>
```

- 모든 사용자 지정 데이터는 dataset 속성을 통해 사용할 수 있다. 이는 돔 조작을 위헤서 html에서 제공해 주는 속성이다.

https://developer.mozilla.org/ko/docs/Web/HTML/Global_attributes/data-*

- 예를 들어 `data-test-value` 라는 이름의 특성을 지정했다면,

  JavaScript에서는 `element.dataset.testValue` 로 접근할 수 있다. 이때 속성명 작성 시 주의사항이 있다.

  속성명 작성 시 주의사항

  - 대소문자 여부에 상관없이 xml로 시작하면 안 됨
  - 세미콜론을 포함해서는 안됨

- 대문자를 포함해서는 안됨

csrftoken 보내기

- 먼저 hidden 타입으로 숨겨져있는 csrf 값을 가진 input 태그를 선택해야 한다.

https://docs.djangoproject.com/en/3.2/ref/csrf/


```
<!-- accounts/profile.html -->

<script>
  const form = document.querySelector('#follow-form')
  form.addEventListener('submit', function(event) {
    event.preventDefault()

    const userId = event.target.dataset.userId
    const csrftoken = document.querySelector('[name=csrfmiddlewaretoken]').value

    axios({
      method: 'post',
      url: `/accounts/${userId}/follow/`
    })

  })
</script>
```

- AJAX로 csfrtoken 보내기

```
<!-- accounts/profile.html -->

<script>
  const form = document.querySelector('#follow-form')
  form.addEventListener('submit', function(event) {
    event.preventDefault()
```

```
    const userId = event.target.dataset.userId
    const csrftoken = document.querySelector('[name=csrfmiddlewaretoken]').value

    axios({
      method: 'post',
      url: `/accounts/${userId}/follow/`,
      headers: {'X-CSRFToken': csrftoken,}
    })

  })
</script>
```

팔로우 버튼 토글하기

- 팔로우 버튼을 토글하기 위해서는 현재 팔로우가 된 상태인지 여부 확인이 필요하다.
- axios 요청을 통해 받는 response 객체를 활용하여 view 함수를 통해서 팔로우 여부를 파악할 수 있도록 변수를 만들고, 그 변수에 팔로우 여부를 저장해서 JSON 타입으로 응답할 것이다.
- 팔로우 여부를 확인하기 위한 is_followed 변수 작성 및 JSON 응답하는 코드를 추가하자.

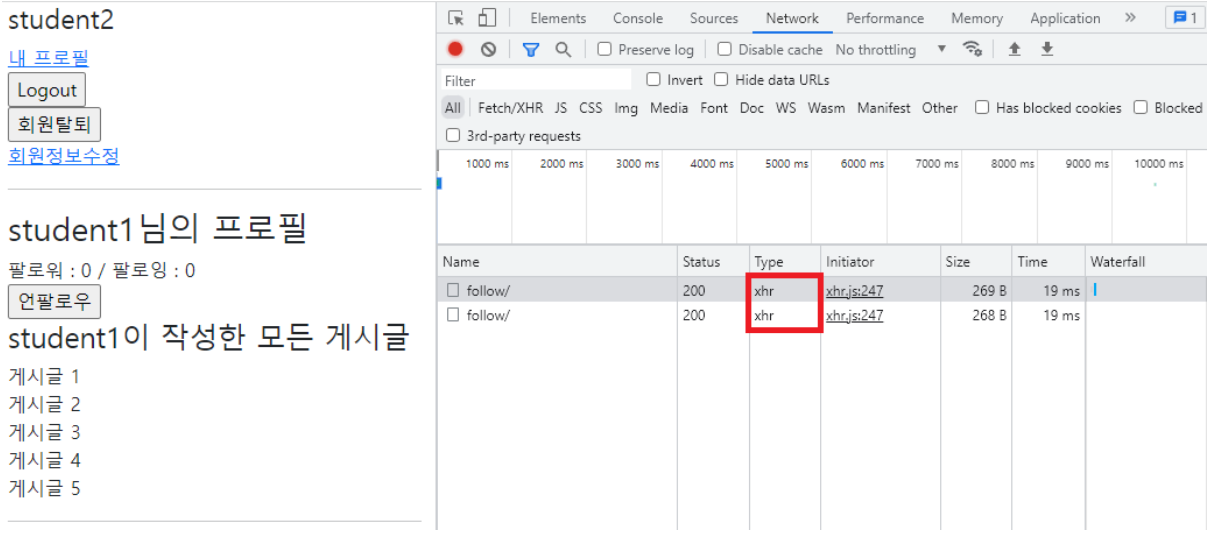
여러분들의 가독성에 도움이 될 수 있도록 you 그리고 me라는 변수로 수정해 보겠다.

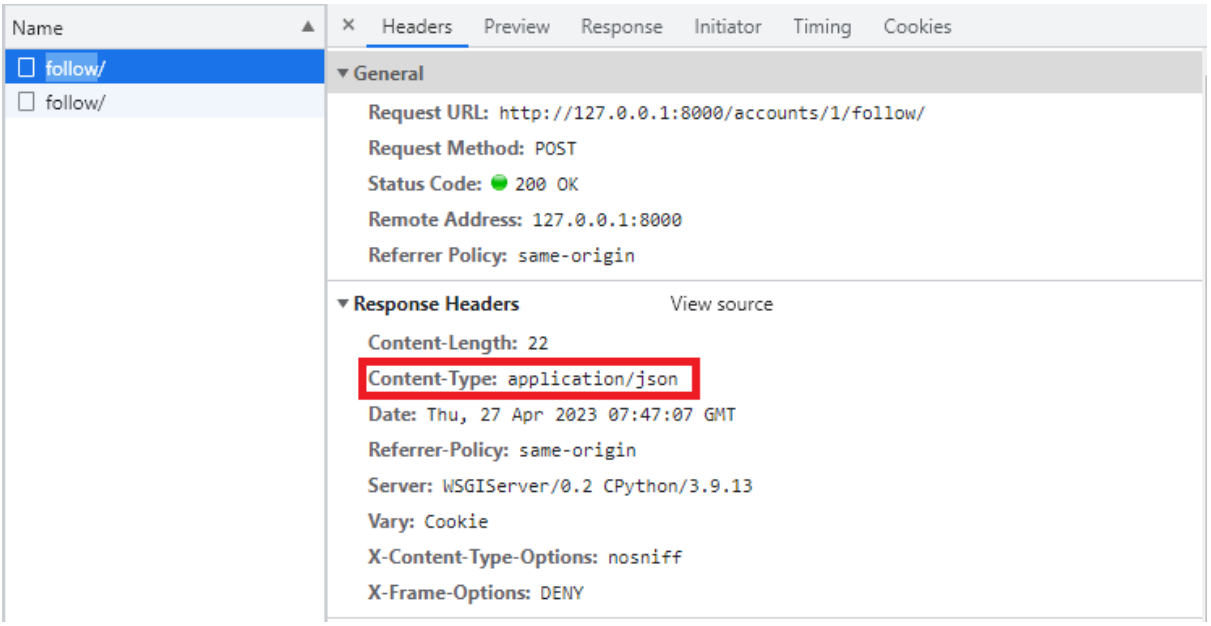
```
# accounts/views.py


@require_POST
# def follow(request, user_pk):
#     if request.user.is_authenticated:
#         User = get_user_model()
#         person = User.objects.get(pk=user_pk)
#         if person != request.user:
#             if person.followers.filter(pk=request.user.pk).exists():
#                 person.followers.remove(request.user)
#             else:
#                 person.followers.add(request.user)
#         return redirect('accounts:profile', person.username)
#     return redirect('accounts:login')



from django.http import JsonResponse

@require_POST
def follow(request, user_pk):
    if request.user.is_authenticated:
        User = get_user_model()
        me = request.user
        you = User.objects.get(pk=user_pk)
        if me != you:
            if you.followers.filter(pk=me.pk).exists():
                you.followers.remove(me)
```

```
                is_followed = False
            else:
                you.followers.add(me)
                is_followed = True
            context = {
                'is_followed': is_followed,
            }
            return JsonResponse(context)
        return redirect('accounts:profile', you.username)
    return redirect('accounts:login')
```

- view 함수에서 응답한 is_followed를 사용해 버튼 토글할 것이다. .then 구문을 추가해 주자

```
<!-- accounts/profile.html -->

<script>
  const form = document.querySelector('#follow-form')
  form.addEventListener('submit', function(event) {
    event.preventDefault()

    const userId = event.target.dataset.userId
    const csrftoken = document.querySelector('[name=csrfmiddlewaretoken]').value

    axios({
      method: 'post',
      url: `/accounts/${userId}/follow/`,
      headers: {'X-CSRFToken': csrftoken,}
    })
      .then((response) => {

        const isFollowed = response.data.is_followed
        const followBtn = document.querySelector('#follow-form > input[type=submit]')

        if (isFollowed === true) {
          // 팔로우 상태일 때는 언팔로우 버튼
          followBtn.value = '언팔로우'
        } else {
          followBtn.value = '팔로우'
        }
      })

  })
</script>
```





팔로워 & 팔로잉 수 비동기 적용

- 해당 요소를 선택할 수 있도록 span 태그와 id 속성 작성

```
<!-- accounts/profile.html -->

{% extends 'base.html' %}

{% block content %}
  <h1>{{ person.username }}님의 프로필</h1>
  <div>
    팔로워 : <span id="followers-count">{{ person.followers.all|length }}</span> /
```

```
    팔로잉 : <span id="followings-count">{{ person.followings.all|length }}</span>
  </div>
```

- 직전에 작성한 span 태그를 각각 선택

```
<!-- accounts/profile.html -->

<script>
  const form = document.querySelector('#follow-form')
  form.addEventListener('submit', function(event) {
    event.preventDefault()

    const userId = event.target.dataset.userId
    const csrftoken = document.querySelector('[name=csrfmiddlewaretoken]').value

    axios({
      method: 'post',
      url: `/accounts/${userId}/follow/`,
      headers: {'X-CSRFToken': csrftoken,}
    })
      .then((response) => {

        const isFollowed = response.data.is_followed
        const followBtn = document.querySelector('#follow-form > input[type=submit]')

        const followersCountTag = document.querySelector('#followers-count')
        const followingsCountTag = document.querySelector('#followings-count')

        if (isFollowed === true) {
          followBtn.value = '언팔로우'
        } else {
          followBtn.value = '팔로우'
        }
      })

  })
</script>
```

- 팔로워, 팔로잉 인원 수 연산은 view 함수에서 진행하여 결과를 응답으로 전달

```
# accounts/views.py

from django.http import JsonResponse

@require_POST
def follow(request, user_pk):
    if request.user.is_authenticated:
        User = get_user_model()
        me = request.user
        you = User.objects.get(pk=user_pk)
        if me != you:
            if you.followers.filter(pk=me.pk).exists():
```

```
                you.followers.remove(me)
                is_followed = False
            else:
                you.followers.add(me)
                is_followed = True
            context = {
                'is_followed': is_followed,
                'followers_count': you.followers.count(),
                'followings_count': you.followings.count(),
            }
            return JsonResponse(context)
        return redirect('accounts:profile', you.username)
    return redirect('accounts:login')
```

- view 함수에서 응답한 연산 결과를 사용해 각 태그의 인원 수 값 변경하기

```
<!-- accounts/profile.html -->

<script>
  const form = document.querySelector('#follow-form')
  form.addEventListener('submit', function(event) {
    event.preventDefault()

    const userId = event.target.dataset.userId
    const csrftoken = document.querySelector('[name=csrfmiddlewaretoken]').value

    axios({
      method: 'post',
      url: `/accounts/${userId}/follow/`,
      headers: {'X-CSRFToken': csrftoken,}
    })
      .then((response) => {

        const isFollowed = response.data.is_followed
        const followBtn = document.querySelector('#follow-form > input[type=submit]')

        const followersCountTag = document.querySelector('#followers-count')
        const followingsCountTag = document.querySelector('#followings-count')

        const followersCount = response.data.followers_count
        const followingsCount = response.data.followings_count

        followersCountTag.innerText = followersCount
        followingsCountTag.innerText = followingsCount

        if (isFollowed === true) {
          followBtn.value = '언팔로우'
        } else {
          followBtn.value = '팔로우'
        }
      })

  })
</script>
```

최종 코드

HTML

```
<!-- accounts/profile.html -->

{% extends 'base.html' %}

{% block content %}
  <h1>{{ person.username }}님의 프로필</h1>
  <div>
    팔로워 : <span id="followers-count">{{ person.followers.all|length }}</span> /
    팔로잉 : <span id="followings-count">{{ person.followings.all|length }}</span>
  </div>

  {% if request.user != person %}
  <div>
    <form id="follow-form" data-user-id="{{ person.pk }}">
      {% csrf_token %}
      {% if request.user in person.followers.all %}
        <input type="submit" value="언팔로우">
      {% else %}
        <input type="submit" value="팔로우">
      {% endif %}
    </form>
  <div>
  {% endif %}
...
```

Python

```
# accounts/views.py

from django.http import JsonResponse

@require_POST
def follow(request, user_pk):
    if request.user.is_authenticated:
        User = get_user_model()
        me = request.user
        you = User.objects.get(pk=user_pk)
        if me != you:
            if you.followers.filter(pk=me.pk).exists():
                you.followers.remove(me)
                is_followed = False
            else:
                you.followers.add(me)
                is_followed = True
            context = {
                'is_followed': is_followed,
                'followers_count': you.followers.count(),
```

```
                'followings_count': you.followings.count(),
            }
            return JsonResponse(context)
        return redirect('accounts:profile', you.username)
    return redirect('accounts:login')
```

JavaScript

```
<!-- accounts/profile.html -->

{% block script %}
<script src="https://cdn.jsdelivr.net/npm/axios/dist/axios.min.js"></script>
<script>
  const form = document.querySelector('#follow-form')
  form.addEventListener('submit', function(event) {
    event.preventDefault()

    const userId = event.target.dataset.userId
    const csrftoken = document.querySelector('[name=csrfmiddlewaretoken]').value

    axios({
      method: 'post',
      url: `/accounts/${userId}/follow/`,
      headers: {'X-CSRFToken': csrftoken,}
    })
      .then((response) => {

        const isFollowed = response.data.is_followed
        const followBtn = document.querySelector('#follow-form > input[type=submit]')

        const followersCountTag = document.querySelector('#followers-count')
        const followingsCountTag = document.querySelector('#followings-count')

        const followersCount = response.data.followers_count
        const followingsCount = response.data.followings_count
        followersCountTag.innerText = followersCount
        followingsCountTag.innerText = followingsCount

        if (isFollowed === true) {
          followBtn.value = '언팔로우'
        } else {
          followBtn.value = '팔로우'
        }
      })

  })
</script>
{% endblock script %}
```

[부록] 부트스트랩 적용하기

- HTML

```
<!-- accounts/profile.html -->

{% extends 'base.html' %}

{% block content %}
  <h1>{{ person.username }}님의 프로필</h1>
  <div>
    팔로워 : <span id="followers-count">{{ person.followers.all|length }}</span> /
    팔로잉 : <span id="followings-count">{{ person.followings.all|length }}</span>
  </div>

  {% if request.user != person %}
  <div>
    <form id="follow-form" data-user-id="{{ person.pk }}">
      {% csrf_token %}
      {% if request.user in person.followers.all %}
        <button type="submit" class="btn btn-secondary">언팔로우</button>
        {% else %}
        <button type="submit" class="btn btn-primary">팔로우</button>
      {% endif %}
    </form>
  <div>
  {% endif %}
```

- JavaScript

```
<!-- accounts/profile.html -->

{% block script %}
<script src="https://cdn.jsdelivr.net/npm/axios/dist/axios.min.js"></script>
<script>
  const form = document.querySelector('#follow-form')
  form.addEventListener('submit', function(event) {
    event.preventDefault()

    const userId = event.target.dataset.userId
    const csrftoken = document.querySelector('[name=csrfmiddlewaretoken]').value

    axios({
      method: 'post',
      url: `/accounts/${userId}/follow/`,
      headers: {'X-CSRFToken': csrftoken,}
    })
      .then((response) => {

        const isFollowed = response.data.is_followed
        const followBtn = document.querySelector('#follow-form > button[type=submit]')

        const followersCountTag = document.querySelector('#followers-count')
        const followingsCountTag = document.querySelector('#followings-count')

        followBtn.classList.toggle('btn-secondary')
        followBtn.classList.toggle('btn-primary')

        const followersCount = response.data.followers_count
        const followingsCount = response.data.followings_count
```

```
        followersCountTag.innerText = followersCount
        followingsCountTag.innerText = followingsCount

        if (isFollowed === true) {
          followBtn.innerText = '언팔로우'
        } else {
          followBtn.innerText = '팔로우'
        }
      })

  })
</script>
{% endblock script %}
```

비동기 적용하기 - 좋아요 (like)

- 좋아요 비동기 적용은 “팔로우와 동일한 흐름 + `forEach()` & `querySelectorAll()`”
  - index 페이지 각 게시글에 좋아요 버튼이 있기 때문

HTML

```
<!-- articles/index.html -->

{% extends 'base.html' %}

{% block content %}
  <h1>Articles</h1>
  {% if request.user.is_authenticated %}
    <a href="{% url 'articles:create' %}">CREATE</a>
  {% endif %}
  <hr>
  {% for article in articles %}
    <p>
      <b>작성자 : <a href="{% url 'accounts:profile' article.user %}">{{ article.user }}</a></b>
    </p>
    <p>글 번호 : {{ article.pk }}</p>
    <p>제목 : {{ article.title }}</p>
    <p>내용 : {{ article.content }}</p>
    <div>
      <form class="like-forms" data-article-id="{{ article.pk }}">
        {% csrf_token %}
        {% if request.user in article.like_users.all %}
          <input type="submit" value="좋아요 취소" id="like-{{ article.pk }}">
        {% else %}
          <input type="submit" value="좋아요" id="like-{{ article.pk }}">
        {% endif %}
      </form>
    </div>
    <a href="{% url 'articles:detail' article.pk %}">상세 페이지</a>
    <hr>
  {% endfor %}
{% endblock content %}
```

Python

```
# articles/views.py

from django.http import JsonResponse

@require_POST
def likes(request, article_pk):
    if request.user.is_authenticated:
        article = Article.objects.get(pk=article_pk)

        if article.like_users.filter(pk=request.user.pk).exists():
            article.like_users.remove(request.user)
            is_liked = False
        else:
            article.like_users.add(request.user)
            is_liked = True
        context = {
            'is_liked': is_liked,
        }
        return JsonResponse(context)
    return redirect('accounts:login')
```

JavaScript

```
{% block script %}
  <script src="https://cdn.jsdelivr.net/npm/axios/dist/axios.min.js"></script>
  <script>
    const forms = document.querySelectorAll('.like-forms')
    const csrftoken = document.querySelector('[name=csrfmiddlewaretoken]').value

      forms.forEach((form) => {
      form.addEventListener('submit', function (event) {
        event.preventDefault()  // form의 기본 기능 막아야 새로고침이 되지 않음

        const articleId = event.target.dataset.articleId

        axios({
          method: 'post',
          url: `http://127.0.0.1:8000/articles/${articleId}/likes/`,
          headers: {'X-CSRFToken': csrftoken},
        })
          .then((response) => {

            const isLiked = response.data.is_liked

            const likeBtn = document.querySelector(`#like-${articleId}`)
            if (isLiked === true) {
              likeBtn.value = '좋아요 취소'
            } else {
              likeBtn.value = '좋아요'
            }
          })
          .catch((error) => {
```

```
            console.log(error.response)
          })
      })
    })
  </script>
{% endblock script %}
```