



# 5차 시험 과제 제출

풀스택 자바 웹개발자 강의  
방미소



# 목차

1. 레이아웃 및 주요기능 정의
2. 구현 결과
3. 문항별 세부 구현 내용 (1~3)

# 1. 레이아웃 및 주요기능 정의

## 카카오 책 검색

정확도순 ▾

10건 ▾

검색어 입력

검색



NO IMAGE  
AVAILABLE

글 제목

본문

저자 | 출판사 | 가격 | 할인가 |

## form 영역

- 옵션값, 검색어를 입력 받는 form

## 주요 HTML

- select 1 : 정확도순 / 발간순 정렬
- select 2 : 한 페이지 검색 건수 설정
- input : 검색어 입력
- button : form submit 이벤트 발생

# 1. 레이아웃 및 주요기능 정의


## 카카오 책 검색

정확도순 ▾

10건 ▾

검색어 입력

검색

  
NO IMAGE  
AVAILABLE

글 제목

본문

저자 | 출판사 | 가격 | 할인가 |

## list 영역

- 결과를 출력할 list

## 주요 HTML

- `img` : 도서 썸네일 이미지를 표시
- `h2` : 도서 제목 표시
- `p` : 도서 설명 표시
- `span` : 저자, 출판사, 가격, 할인가 등 표시

## 2. 구현 결과

### 카카오 책 검색

정확도순 ▾ 10건 ▾ 테스트 x 검색



#### 위풍당당 여우 꼬리. 2: 알쏭달쏭 우정 테스트(양장본 HardCover)

어쩐지 마음에 안 드는 윤나가 단미에게 불쑥 초대장을 보낸다. 하필 단짝 친구 루미와의 사이가 멀렁거리는 이때에 무슨 일일까? 어긋난 관계 때문에 세상이 무너지고, 외롭고, 쓸쓸하고, 화가 나고, 답답한 느낌만 가득한데 말이다! 단미는 수상한 우정 ...

손원평 | 창비 | 12000 | 10800 |



#### 해커스 보카 중학 고난도 + 미니암기장 + 누적 테스트북 세트

있는 주제별 구성의 연상 학습 효과를 통해 단어를 더 쉽고 빠르게 외울 수 있습니다. 단어의 뜻을 이미지화 한 'Picture review'를 통해 재미있게 학습할 수 있을 뿐만 아니라, 기술 문장을 활용한 '예문'과 중간/기말 시험의 문제유형이 반영된 'Daily Test'로...

해커스 어학연구소 | 해커스어학연구소 | 11000 | 9900 |



#### Do it! 알고리즘 코딩 테스트: 자바 편

"코딩 테스트는 어떻게 준비해야 할까?" 곧 코딩 테스트를 앞두고 있거나 올해 안에 IT 기업으로 취업 또는 이직을 준비하고 있다면 누구나 이런 고민을 할 것이다. <Do it! 알고리즘 코딩 테스트? 자바 편>에 그 답이 있다. 개발 12년, 강의 5년 동안 쌓은...

김종관 | 이지스퍼블리싱 | 32000 | 28800 |



#### 단위 테스트

소프트웨어 개발에 있어 단위 테스트는 이제 선택이 아니라 필수가 됐다. 단위 테스트에 대한 오해를 바로잡고, 올바른 단위 테스트에 대한 원칙, 테스트를 작성하는 스타일과 효과적인 테스트를 위한 소프트웨어 아키텍처를 이해할 수 있다. 또한 단위 테...

블라디미르 코리코프 | 에이콘출판 | 35000 | 31500 |



#### 해커스 보카 중학 필수 + 미니암기장 + 누적 테스트북 세트

있는 주제별 구성의 연상 학습 효과를 통해 단어를 더 쉽고 빠르게 외울 수 있습니다. 단어의 뜻을 이미지화 한 'Picture review'를 통해 재미있게 학습할 수 있을 뿐만 아니라, 기술 문장을 활용한 '예문'과 중간/기말 시험의 문제유형이 반영된 'Daily Test'로...

### 카카오 책 검색

발간일순 ▾ 50건 ▾ 코딩 검색



#### 생활코딩! HTML+CSS+자바스크립트(위키북스 러닝스쿨 시리즈...)

최신 방법과 Web Server for Chrome을 활용해 웹 서버를 가장 간편하게 구축하는 방법, 내가 만든 웹 페이지에 간단한 코드만으로 동영상 삽입, 댓글, 채팅, 방문자 분석 기능을 추가하는 법을 소개합니다. ★ 이 책에서 다루는 내용 ★ © 코딩이란 무엇인...

이고잉 | 위키북스 | 28000 | 25200 |



#### 인공지능 자율주행 RC카 체험하고 코딩하기 with 라즈베리파이

<인공지능 자율주행 RC카로 체험하고 코딩하기 with 라즈베리파이> 책은 인공지능 자율주행을 짧은 시간에 체험할 수 있으며 원한다면 인공지능 자율주행 자동차의 내부를 직접 코딩하며 인공지능과 관련된 공부를 심도있게 수행할 수 있습니다. 책은 ...

서민우 | 앤서북 | 18000 | 16200 |



#### Do it! 알고리즘 코딩 테스트: 자바 편

"코딩 테스트는 어떻게 준비해야 할까?" 곧 코딩 테스트를 앞두고 있거나 올해 안에 IT 기업으로 취업 또는 이직을 준비하고 있다면 누구나 이런 고민을 할 것이다. <Do it! 알고리즘 코딩 테스트? 자바 편>에 그 답이 있다. 개발 12년, 강의 5년 동안 쌓은...

김종관 | 이지스퍼블리싱 | 32000 | 28800 |



#### 생활코딩! 자바 프로그래밍 입문

이고잉 | 위키북스 | 20000 | 할인가 없음 |



#### Do it! 알고리즘 코딩 테스트 - 자바 편

IT 기업 취업과 이직의 필수 단계인 알고리즘 코딩 테스트! 출제 경향을 완벽하게 반영한 핵심 100제로 한 번에 합격한다! "코딩 테스트는 어떻게 준비해야 할까?" 곧 코딩 테스트를 앞두고 있거나 올해 안에 IT 기업으로 취업 또는 이직을 준비하고 있다...

김종관 | 이지스퍼블리싱 | 22000 | 할인가 없음 |

### 3. 문항별 세부 구현 내용 - 문항 1

kakao developers

내 애플리케이션

제품

더 보기 ▾

miso\_9207@naver.com ▾

Q

≡

내 애플리케이션 > 앱 설정 > 요약 정보

APP

검색기능 연습

ID 726787

OWNER

앱 키

|              |                                  |
|--------------|----------------------------------|
| 네이티브 앱 키     | 06276faf1a0d51c887e5c9322c1bf554 |
| REST API 키   | 71676e7d7d63ac9908e5bd4166df7109 |
| JavaScript 키 | c4306ace4c84d87237912d2c0d85fdd9 |
| Admin 키      | d95d58e4a8466539bec9e88c33ef834c |

플랫폼

설정된 플랫폼 정보가 없습니다. [플랫폼 설정하기](#)

기본 정보

|      |         |
|------|---------|
| 앱 ID | 726787  |
| 앱 이름 | 검색기능 연습 |

카카오 OpenAPI 인증키 발급 화면

- REST API키 등 확인 가능

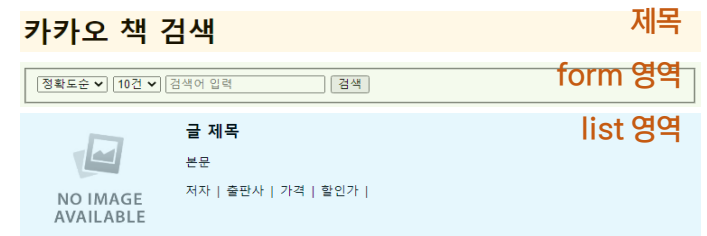
### 3. 문항별 세부 구현 내용 - 문항 2

```
<body>
<div id="loading"></div>
<h1>카카오 책 검색</h1>

<form id="searchForm">
  <fieldset>
    <!-- select1 -->
    <select name="sort" id="sort">
      <option value="accuracy">정확도순</option>
      <option value="latest">발간일순</option>
    </select>
    <!-- select2 -->
    <select name="size" id="size">
      <option value="10">10건</option>
      <option value="20">20건</option>
      <option value="30">30건</option>
      <option value="40">40건</option>
      <option value="50">50건</option>
    </select>
    <!-- query input -->
    <input type="search", id="query", placeholder="검색어 입력">
    <button type="submit">검색</button>
  </fieldset>
</form>

<ul id="bookList">
  <li>
    <a href="#" class="flex-row">
      <div class="img-con">
        
      </div>
      <div class="text-con">
        <h2>글 제목</h2>
        <p>본문</p>
        <span>저자</span>
        <span>출판사</span>
        <span>가격</span>
        <span>할인가</span>
      </div>
    </a>
  </li>
</ul>

<script src="https://cdn.jsdelivr.net/npm/axios/dist/axios.min.js"></script>
<script src="./libs/searchHelper.js"></script>
</body>
```



## HTML DOM Tree

### ① loading 아이콘

- <div> 태그로 영역을 설정하여 CSS로 배경이미지를 넣어준 후, display:none으로 설정해 특정 상황에서 표시되도록 함
- position은 항상 화면 중앙에 오도록 설정

```
#loading {
  width: 100px;
  height: 100px;
  background: url(img/loading.gif) no-repeat;
  background-size: cover;
  background-position: center center;
  position: fixed;
  left: 50%;
  top: 50%;
  transform: translate(-50%, -50%);
  z-index: 9999;
  display: none;
}
```

### ② 제목

- <h1> 태그로 페이지 타이틀 표현

### 3. 문항별 세부 구현 내용 - 문항 2

```
<body>
<div id="loading"></div>
<h1>카카오 책 검색</h1>

<form id="searchForm">
  <fieldset>
    <!-- select1 -->
    <select name="sort" id="sort">
      <option value="accuracy">정확도순</option>
      <option value="latest">발간일순</option>
    </select>
    <!-- select2 -->
    <select name="size" id="size">
      <option value="10">10건</option>
      <option value="20">20건</option>
      <option value="30">30건</option>
      <option value="40">40건</option>
      <option value="50">50건</option>
    </select>
    <!-- query input -->
    <input type="search", id="query", placeholder="검색어 입력">
    <button type="submit">검색</button>
  </fieldset>
</form>

<ul id="bookList">
  <li>
    <a href="#" class="flex-row">
      <div class="img-con">
        
      </div>
      <div class="text-con">
        <h2>글 제목</h2>
        <p>본문</p>
        <span>저자</span>
        <span>출판사</span>
        <span>가격</span>
        <span>할인가</span>
      </div>
    </a>
  </li>
</ul>

<script src="https://cdn.jsdelivr.net/npm/axios/dist/axios.min.js"></script>
<script src="./libs/searchHelper.js"></script>
</body>
```

form 영역

카카오 책 검색

제목

정확도순 10건 검색어 입력

검색

form 영역



NO IMAGE  
AVAILABLE

글 제목

본문

저자 | 출판사 | 가격 | 할인가 |

list 영역

## HTML DOM Tree

### ③ form 영역

- <fieldset> : 연관 폼 요소들을 묶음으로 정의
- <select> 1 : 정렬방식을 설정 / option 값을 JS에 전달
- <select> 2 : 표시 검색 건수를 설정 / option 값을 JS에 전달
- <input> : type은 search로 설정 / 입력된 text를 JS에 전달
- <button> : type은 submit으로 설정 / JS에서 이벤트 발생



### 3. 문항별 세부 구현 내용 - 문항 2

```
<body>
  <div id="loading"></div>
  <h1>카카오 책 검색</h1>

  <form id="searchForm">
    <fieldset>
      <!-- select1 -->
      <select name="sort" id="sort">
        <option value="accuracy">정확도순</option>
        <option value="latest">발간일순</option>
      </select>
      <!-- select2 -->
      <select name="size" id="size">
        <option value="10">10건</option>
        <option value="20">20건</option>
        <option value="30">30건</option>
        <option value="40">40건</option>
        <option value="50">50건</option>
      </select>
      <!-- query input -->
      <input type="search", id="query", placeholder="검색어 입력">
      <button type="submit">검색</button>
    </fieldset>
  </form>

  <ul id="bookList">
    <li>
      <a href="#" class="flex-row">
        <div class="img-con">
          
        </div>
        <div class="text-con">
          <h2>글 제목</h2>
          <p>본문</p>
          <span>저자</span>
          <span>출판사</span>
          <span>가격</span>
          <span>할인가</span>
        </div>
      </a>
    </li>
  </ul>

  <script src="https://cdn.jsdelivr.net/npm/axios/dist/axios.min.js"></script>
  <script src="./libs/searchHelper.js"></script>
</body>
```

list 영역

카카오 책 검색

제목

정확도순 10건 검색어 입력

검색

form 영역



NO IMAGE AVAILABLE

글 제목

본문

저자 | 출판사 | 가격 | 할인가 |

list 영역

## HTML DOM Tree

### ④ list 영역

- <ul> : id를 booklist로 설정
- <li> : 하나의 검색 건수가 담길 영역
- <a> : 영역을 클릭했을 때 관련글로 이동하도록 링크 부여
- <div> : 각각 이미지, 텍스트를 담는 박스를 만들어 flex 정렬
- <img> : 기본적으로 no image이며 책 이미지가 있으면 표기
- <h2> : 책 타이틀이므로 h 태그로 강조
- <p> : 책 설명 내용이 들어갈 부분
- <span> : 각각 저자, 출판사, 가격, 할인가 정보를 담을 부분

### 3. 문항별 세부 구현 내용 - 문항 3

```
/**
 * @filename: searchHelper.js
 * @author: 방미소
 * @description: 카카오 검색 API를 활용하여 입력된 값에 대한 검색 결과를 출력한다.
 */

let currentPage = 1;
let isEnd = false;
let queryKeyword, querySort, querySize;

const KAKAO_REST_KEY = '71676e7d7d63ac9908e5bd4166df7109';
```

#### Axios 라이브러리 활용 구현 결과

##### ① 공통 영역

- 기본 변수 선언 / API 키 세팅

### 3. 문항별 세부 구현 내용 - 문항 3

```
const letSearch = async () => {
  // 로딩
  const loading = document.querySelector('#loading');
  loading.classList.add('active');

  // 페이지 초기화
  const booklist = document.querySelector('#bookList')
  if (currentPage === 1) { ...
  }

  // API 전달 파라미터 설정
  queryKeyword = document.querySelector('#query').value;
  querySort = document.querySelector('#sort').value;
  querySize = document.querySelector('#size').value;

  const K_URL = 'https://dapi.kakao.com/v3/search/book';
  const K_VALUE = { ...
  };

  // Axios
  let json = null;

  try { ...
  }
  catch (err) { ...
  }
  finally { ...
  }

  if (json !== null) { ...
  }
}
```

#### Axios 라이브러리 활용 구현 결과

##### ② 검색 함수 영역

- async 형태로 구현하여 다른 작업과 별개로 병렬처리
- 함수 실행 직후 loading 아이콘이 display:block 전환되며 화면 정중앙에 나타남 (CSS 클래스 추가로 구현)
- 이후 기존 검색결과가 담긴 <ul> 태그가 초기화되며 모든 <li> 삭제
- 이후 Axios 라이브러리를 통해 유효 URL에 접속, 유효 key와 파라미터를 전달하고 반환되는 데이터를 json 형태로 받아옴

### 3. 문항별 세부 구현 내용 - 문항 3

```
// 로딩
const loading = document.querySelector('#loading');
loading.classList.add('active');

// 페이지 초기화
const booklist = document.querySelector('#bookList')
if (currentPage === 1) {
  Array
    .from(booklist.getElementsByTagName('li'))
    .forEach(v => {
      booklist.removeChild(v);
    });
}
```

#### Axios 라이브러리 활용 구현 결과

##### ② 검색 함수 영역 (상세)

- 로딩 : id="loading"인 <div> 를 가져와 active 클래스 추가
- 페이지 초기화 : 현재 페이지가 1페이지라면, <ul> 내의 <li> 목록을 가져온 후(html collection) 배열로 변환하여 forEach 태그 적용
  - 이 조건은 신규 검색을 실행하는 submit 이벤트에서 현재 페이지를 1로 설정하면서 작동함
  - 신규 검색이 아닌 일반 스크롤 이벤트에서 작동하지 않음

### 3. 문항별 세부 구현 내용 - 문항 3

```
// API 전달 파라미터 설정
queryKeyword = document.querySelector('#query').value;
querySort = document.querySelector('#sort').value;
querySize = document.querySelector('#size').value;

const K_URL = 'https://dapi.kakao.com/v3/search/book';
const K_VALUE = {
  params: {
    query: queryKeyword,
    sort: querySort,
    size: querySize,
    page: currentPage
  },
  headers: {
    Authorization: `KakaoAK ${KAKAO_REST_KEY}`
  }
};

// Axios
let json = null;

try {
  json = await axios.get(K_URL, K_VALUE);
  console.log(json);
} catch (err) {
  alert('요청 처리 중 오류가 발생하였습니다. ');
  return;
} finally {
  loading.classList.remove('active');
}
```

#### Axios 라이브러리 활용 구현 결과

##### ② 검색 함수 영역 (상세)

- 최상단에 선언했던 변수들에 적절한 값 부여
  - selection, input 태그에서 맞는 value를 가져옴
- URL과 파라미터를 올바른 형태로 설정 후 Axios에 전달
- 통신 결과로 가져올 데이터를 담을 변수 'json' 초기화

### 3. 문항별 세부 구현 내용 - 문항 3

```
if (json !== null) {
  const {data} = json;
  isEnd = data.meta.is_end;

  data.documents.map(v => {
    const div1 = document.createElement('div');
    const div2 = document.createElement('div');
    const li = document.createElement('li');
    const a = document.createElement('a');
    const img = document.createElement('img');
    const h2 = document.createElement('h2');
    const p = document.createElement('p');
    const span1 = document.createElement('span');
    const span2 = document.createElement('span');
    const span3 = document.createElement('span');
    const span4 = document.createElement('span');

    a.setAttribute('class', 'flex-row');
    a.setAttribute('href', v.url);
    div1.setAttribute('class', 'img-con');
    div2.setAttribute('class', 'text-con');
    img.setAttribute('src', v.thumbnail || 'img/noimage.jpg');

    h2.innerHTML = v.title;
    p.innerHTML = v.contents;
    span1.innerHTML = v.authors;
    span2.innerHTML = v.publisher;
    span3.innerHTML = v.price;
    span4.innerHTML = (v.sale_price === -1 ? '할인가 없음' : v.sale_price);

    div1.appendChild(img);
    div2.appendChild(h2);
    div2.appendChild(p);
    div2.appendChild(span1);
    div2.appendChild(span2);
    div2.appendChild(span3);
    div2.appendChild(span4);
    a.appendChild(div1);
    a.appendChild(div2);
    li.appendChild(a);
    booklist.appendChild(li);
  })
}
```

## Axios 라이브러리 활용 구현 결과

### ② 검색 함수 영역 (상세)

- 오류가 발생하지 않고, 변수 'json'에 정상적으로 데이터가 담겼을 경우에 한해 검색결과 출력
- 'json' 객체 안의 'data' 키 값을 구조분해할당으로 가져온 후, 내부의 documents(실제 도서 정보가 담긴 부분) 요소를 순환하며 알맞게 요소를 생성 / 배치하는 함수 구현
- 'isEnd' 변수에 'data' 값 내의 마지막 페이지 정보 할당
  - 특정 영역의 경우, 가져온 데이터가 없을 경우에 대해 대체 이미지/텍스트 부여 (책 썸네일 없음 / 할인가 정보 없음)

### 3. 문항별 세부 구현 내용 - 문항 3

```
// 신규 검색
document.querySelector('#searchForm').addEventListener('submit', e => {
  e.preventDefault();

  const queryField = document.querySelector('#query');
  queryKeyword = queryField.value.trim();
  console.log(queryKeyword);

  if(!queryKeyword) {
    alert('검색어를 입력하세요.');
```

```
    queryField.focus();
    return;
  }

  currentPage = 1;
  letSearch();
})

// 추가 검색 (스크롤)
window.addEventListener('scroll', e => {
  const loading = document.querySelector('#loading');
  if(isEnd || loading.classList.contains('active')) {
    return;
  }

  const scrollTop = window.scrollY;
  const windowHeight = window.screen.availHeight;
  const documentHeight = document.body.scrollHeight;

  if (scrollTop + windowHeight >= documentHeight) {
    currentPage++;
    letSearch();
  }
})
```

#### Axios 라이브러리 활용 구현 결과

##### ③ 검색 함수 영역

- 해당 함수를 활용하여 button을 눌렀을 때 발생하는 신규검색과 스크롤 시 발생하는 추가검색 기능 구현
- 신규 검색 : preventDefault 메서드로 submit이 발생하는 것을 중단하고, input 박스에 입력된 문자를 가져와 'queryKeyword' 변수에 할당
  - 검색어가 없을 경우 경고창을 띄운 후 input 박스로 포커스를 이동시켜 검색어가 입력되도록 함
- 추가 검색 : 현재 스크롤 위치, 브라우저 화면 높이, 전체 문서 높이를 계산하여 최하단까지 스크롤되었을 경우 추가 검색 실행
  - 'currentPage' 변수에 값을 할당, 파라미터로 전달해 다음 페이지가 검색되도록 함