코드 분석

const url = "https://en.wikipedia.org/wiki/ISO\_3166-1\_alpha-3";

async function scrapeData() {

*try* {

*// Fetch HTML of the page we want to scrape*

const { data } = *await* axios.get(url);

우선 우리가 scraping할 data가 있는 page의 url을 url변수에 넣습니다. 그리고 axios로 비동기적으로 데이터를 받아오기 위해 호출 할 함수를 async로 선언해 줍니다. 그리고 해당 url을 활용해 axios로 받아온 html문서의 data를 구조 분해 할당으로 data에 그 값을 저장해 줍니다.

const $ = cheerio.load(data);

그 다음에는 cheerio를 사용하기 위해 load메소드 안에 scraping할 데이터를 넣고 이의 결괏값을 $로 저장합니다.

const listItems = $(".plainlist ul li");

텍스트이(가) 표시된 사진

자동 생성된 설명

또한 우리가 원하는 데이터는 div.plainlist > ul > li안에 있으므로 이에 해당하는 css selector를 $()의 매개변수 안에 넣고 이에 해당하는 값을 listItems에 저장합니다.

const countries = [];

그리고 countries의 정보를 저장할 배열을 선언해 줍니다.

listItems.each((idx, el) => {

*// Object holding data for each country/jurisdiction*

const country = { name: "", iso3: "" };

*// Select the text content of a and span elements*

*// Store the textcontent in the above object*

country.name = $(el).children("a").text();

country.iso3 = $(el).children("span").text();

*// Populate countries array with country data*

countries.push(country);

});

그리고 아까 위에서 받아온 listItems를 순회하면서 각각의 item을 el이라고 했을 때 { name: “”, iso3: “” }에 맞는 형식으로 el의 데이터를 빼와야 합니다. 그래서 다시 cheerio의 $()매개변수에 el을 넣습니다.

텍스트이(가) 표시된 사진

자동 생성된 설명

그리고 li > span에 el의 iso3에 해당하는 값이, li > a에 el의 name에 해당하는 값이 들어있으므로,

Iso3의 값을 $(el).children(‘a’).text()로 li > a DOM element의 textContent를 추출합니다. 그리고 동일하게 name도 $(el).children(‘span’).text()로 li > span DOM element의 textContent를 추출합니다. 그리고 해당하는 값을 아까 만든 형식의 name, iso3키 값에 저장합니다.

마지막으로 아까 선언한 모든 나라 데이터를 저장할 countries배열에다가 순회하면서 아까 만든 object형식을 하나씩 추가해 갑니다.

console.dir(countries);

그리고 객체의 data를 확인하기 위해 console.dir로 확인합니다.

fs.writeFile("countries.json", JSON.stringify(countries, null, 2), (err) => {

*if* (err) {

console.error(err);

*return*;

}

});

마지막으로 countries.json이라는 파일에 countries배열 object를 indent간격을 2로 정렬과 동시에 string으로 바꾸어 준 값을 저장합니다. 만약 해당 과정에서 promise가 reject되면 error문구를 출력하고 넘어갑니다.

} *catch* (err) {

console.error(err);

}

}

scrapeData();

마지막으로 scrapData함수는 비동기 함수이므로 promise가 reject되는 경우를 위해 try catch문으로 예외를 잡아주고 예외가 발생하면 이를 출력합니다. 그리고 마지막으로 scrapData()로 함수를 호출하는 과정을 거칩니다.

이로서 해당 웹 페이지의 나라 data를 scraping하는 코드 분석이 끝났습니다.

Axios, cheerio, pretty 모듈의 간략한 설명

**< Axios >**

Axios는 브라우저, node.js를 위한 promise API를 활용하는 HTTP비동기 통신 라이브러리 입니다. 쉽게 말해서 백엔드랑 프론트엔드랑 통신을 쉽게 하기 위해 Ajax와 더불어 사용합니다. 이미 자바스크립트에는 fetch api가 있지만, 프레임워크 등에서는 ajax를 구현할 땐 axios를 쓰는 편이라고 보면 됩니다.

또한 운영환경에 따라 브라우저의 XMLHttpRequest객체 또는 Node.js의 http api를 사용합니다. 또한 Promise(ES6) API를 사용하고, 요청과 응답 데이터를 변형하고, HTTP요청 취소와, 요청과 응답을 JSON형태로 자동 변경합니다.

Axios의 사용법에는 다양한 form이 존재하지만 간단하게 많이 사용되는 형태는 아까도 봤지만

axios.[request protocol type](url)이 됩니다. 이 외에도 axios({ method: ‘get’, url: ‘~’}).then(console.log)와 같이 axios의 매개변수로 옵션 변수를 주고 then chain으로 결괏값을 받는 방법이 있습니다.

**< cheerio >**

간단하게 말해서 NodeJS에서 HTML을 파싱하여서 Scrapping하기 위한 라이브러리입니다. 해당 웹사이트에서 파싱할 부분을 정하고, html태그를 찾아내기 위해서 jquery와 같은 문법으로 selector로 선택해서 파싱하는 과정을 거칩니다.

이의 간단한 사용 방법은 아까 예제에서도 봤지만

const cheerio = require('cheerio');

const $ = cheerio.load('<h2 class="title">Hello world</h2>');

$('h2.title').text('Hello there!');

$('h2').addClass('welcome');

$.html();

//=> <html><head></head><body><h2 class="title welcome">Hello there!</h2></body></html>

위와 같이 cheerio모듈을 불러오고 cheerio.load의 매개변수에 html string을 넣고 $를 반환값으로 받습니다. 그 다음에 $()매개변수로 css selector를 통해 우리가 활용하고 싶은 data들을 추출하거나 변형해서 그 값을 html string으로반환합니다.

**< pretty>**

Pretty는 npm공식문서에 some tweaks for beautifying HTML with js-beautify according to my preference라고 나와있습니다. 간단한 예시를 들면

<!DOCTYPE html> <html **lang**="en"> <head>

<meta **charset**="UTF-8"> <title>Home</title>

</head> <body> This is content. </body> </html>

위와 같은 HTM문서를

<!DOCTYPE html>

  <html **lang**="en">

  <head>

    <meta **charset**="UTF-8">

    <title>Home</title>

  </head>

  <body>

    This is content.

  </body>

</html>

아래와 같이 깔끔하게 출력하는 역할을 하는 것 같습니다.

이의 간단한 옵션들로 사용자화 할 수도 있습니다. ( 간격… )

이의 간단한 ocd는

pretty(STRING\_OF\_HTML, {ocd**:** true});

넣고 싶은 html문서를 string화 한 값과 그 뒤에 옵션 값을 넣으면 됩니다. 그럼 이의 반환값이 pretty의 결괏값이 됩니다.