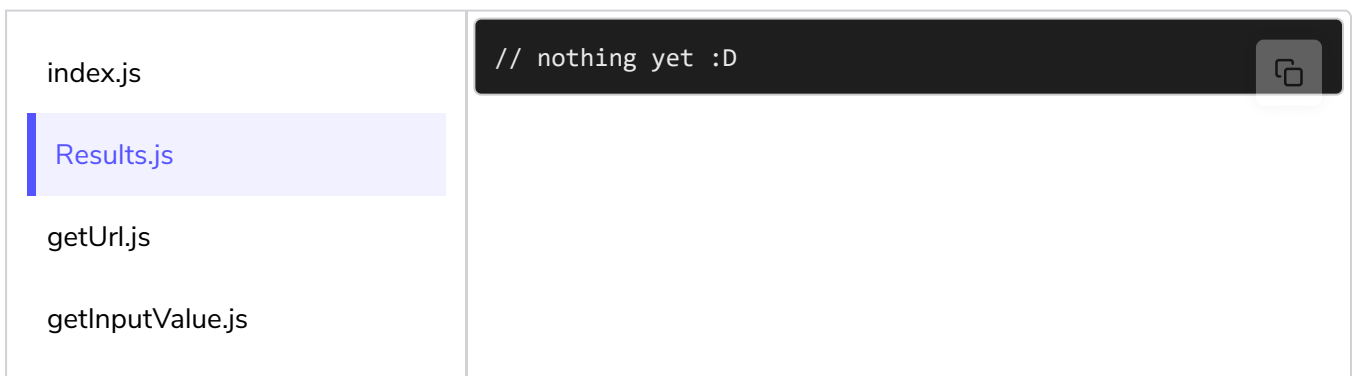


Making a Results Component

Making a component that nicely formats our Wikipedia results. (6 min. read)

Now that we have JSON, let's create a component that pretties it up!

Add **Results.js** to your directory.



Remember our Wikipedia search JSON's shaped like this

```
[
  "JavaScript",
  [
    "JavaScript",
    "JavaScript syntax",
    "JavaScript engine",
    "JavaScript library",
    "JavaScript InfoVis Toolkit",
    "JavaScript Style Sheets",
    "JavaScriptMVC",
    "JavaScript templating",
    "JavaScript framework",
    "JavaScript OSA"
  ],
  [
    "JavaScript (), often abbreviated as JS, is a high-level, interpreted programming language",
    "The syntax of JavaScript is the set of rules that define a correctly structured JavaScript",
    "A JavaScript engine is a program or interpreter which executes JavaScript code. A JavaScript",
    "A JavaScript library is a library of pre-written JavaScript which allows for easier deve",
    "The JavaScript InfoVis Toolkit provides tools for creating Interactive Data Visualizatio",
    "JavaScript Style Sheets (JSSS) was a stylesheet language technology proposed by Netscape",
    "JavaScriptMVC is an open-source rich Internet application framework based on jQuery and",
    "JavaScript templating refers to the client side data binding method implemented with the",
    "A JavaScript framework is an application framework written in JavaScript. It differs fro",
    "JavaScript OSA, (originally JavaScript for OSA, abbreviated as JSOSA), is a freeware int",
  ],
  [
```

```
[
  "https://en.wikipedia.org/wiki/JavaScript",
  "https://en.wikipedia.org/wiki/JavaScript_syntax",
  "https://en.wikipedia.org/wiki/JavaScript_engine",
  "https://en.wikipedia.org/wiki/JavaScript_library",
  "https://en.wikipedia.org/wiki/JavaScript_InfoVis_Toolkit",
  "https://en.wikipedia.org/wiki/JavaScript_Style_Sheets",
  "https://en.wikipedia.org/wiki/JavaScriptMVC",
  "https://en.wikipedia.org/wiki/JavaScript_templating",
  "https://en.wikipedia.org/wiki/JavaScript_framework",
  "https://en.wikipedia.org/wiki/JavaScript_OSA"
]
```

It's an array. Here's the indices

0. Query (what you searched for)
1. Array of result names
2. Array of summaries
3. Array of links to results

Our component can take an array of this shape and return a nicely formatted list.

Edit `Results.js`

index.js

Results.js

getUrl.js

getInputValue.js

```
export default ([query, names, summaries, links]) =>
  <h2>Searching for "${query}"</h2>
  <ul class="list-group">
    ${names.map(
      (name, index) => `
        <li class="list-group-item">
          <a href=${links[index]} target="_blank">
            <h4>${name}</h4>
          </a>
          <p>${summaries[index]}</p>
        </li>
      `
    )}
  </ul>
`;
```

Let's go step by step.

- It's a function that takes an array of our expected elements: `query`, `names`, `summaries`, and `links`.
- Using [ES6 template literals](#), it returns an HTML string with a title and a list.

- Inside the `` we map `names` to `` tags, so one for each.
- Inside those are `<a>` tags pointing to each result's link. Each link opens in a new tab.
- Below the link is a paragraph summary.

Import this in `index.js` and use it like so

index.js

Results.js

getUrl.js

getInputValue.js

```
import 'bootstrap/dist/css/bootstrap.min.css';
import { pipe, tap } from 'ramda';
import getInputValue from './getInputValue';
import getUrl from './getUrl';
import Results from './Results';

const searchAndRenderResults = pipe(
  getInputValue,
  getUrl,
  url =>
    fetch(url)
      .then(res => res.json())
      .then(Results)
      .then(console.warn)
);

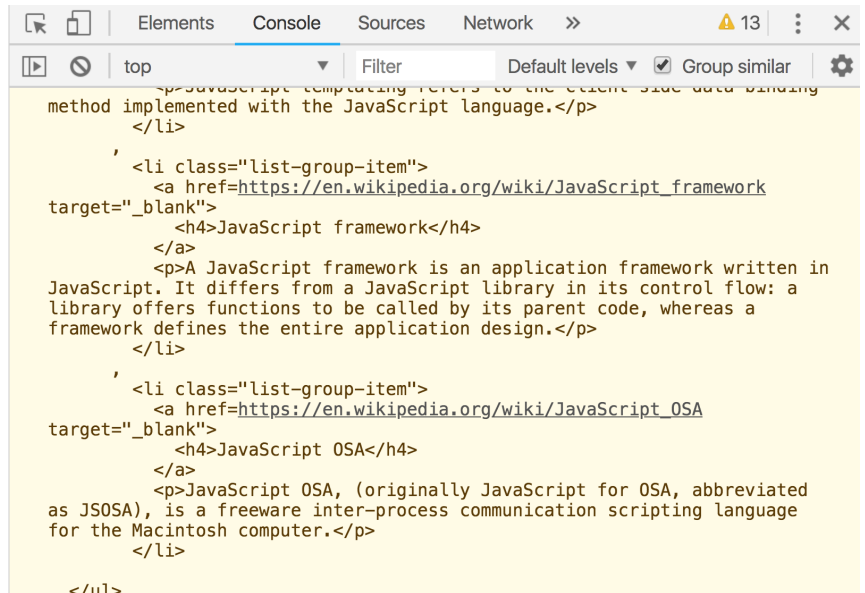
const inputElement = document.querySelector('input');

inputElement.addEventListener('keyup', searchAndRenderResults);
```

Pass the JSON straight to `Results` and log the result.

Wikipedia API Search

JavaScript



The screenshot shows a web browser's developer console with the 'Console' tab selected. It displays HTML markup for search results. The first result is for 'JavaScript framework', and the second is for 'JavaScript OSA'. The markup includes links to Wikipedia pages and descriptive text for each.

```
<p>JavaScript computing refers to the client-side data-binding method implemented with the JavaScript language.</p>
</li>
'
<li class="list-group-item">
  <a href="https://en.wikipedia.org/wiki/JavaScript_framework" target="_blank">
    <h4>JavaScript framework</h4>
  </a>
  <p>A JavaScript framework is an application framework written in JavaScript. It differs from a JavaScript library in its control flow: a library offers functions to be called by its parent code, whereas a framework defines the entire application design.</p>
</li>
'
<li class="list-group-item">
  <a href="https://en.wikipedia.org/wiki/JavaScript_OSA" target="_blank">
    <h4>JavaScript OSA</h4>
  </a>
  <p>JavaScript OSA, (originally JavaScript for OSA, abbreviated as JSOSA), is a freeware inter-process communication scripting language for the Macintosh computer.</p>
</li>
</ul>
```

Pretty cool, we've got our markup! Now let's render it. Add a `render` function to your `index.js`.

index.js

Results.js

getUrl.js

getInputValue.js



```
import 'bootstrap/dist/css/bootstrap.min.css';
import { pipe, tap } from 'ramda';
import getInputValue from './getInputValue';
import getUrl from './getUrl';
import Results from './Results';

const render = markup => {
  const resultsElement = document.getElementById('results');

  resultsElement.innerHTML = markup;
};

const searchAndRenderResults = pipe(
  getInputValue,
  getUrl,
  url =>
    fetch(url)
      .then(res => res.json())
      .then(Results)
      .then(console.warn)
);

const inputElement = document.querySelector('input');

inputElement.addEventListener('keyup', searchAndRenderResults);
```

And replace `console.warn` with `render` in your fetch.

index.js

Results.js

getUrl.js

getInputValue.js



```
import 'bootstrap/dist/css/bootstrap.min.css';
import { pipe, tap } from 'ramda';
import getInputValue from './getInputValue';
import getUrl from './getUrl';
import Results from './Results';

const render = markup => {
  const resultsElement = document.getElementById('results');

  resultsElement.innerHTML = markup;
};

const searchAndRenderResults = pipe(
  getInputValue,
  getUrl,
  url =>
    fetch(url)
      .then(res => res.json())
      .then(Results)
      .then(render)
);

const inputElement = document.querySelector('input');

inputElement.addEventListener('keyup', searchAndRenderResults);
```

Check it out!

Wikipedia API Search

Searching for "JavaScript"

[JavaScript](#)

JavaScript ([link](#)), often abbreviated as JS, is a high-level, interpreted programming language. It is a language which is also characterized as dynamic, weakly typed, prototype-based and multi-paradigm.

[JavaScript syntax](#)

The syntax of JavaScript is the set of rules that define a correctly structured JavaScript program.

[JavaScript engine](#)

A JavaScript engine is a program or interpreter which executes JavaScript code. A JavaScript engine may be a traditional interpreter, or it may utilize just-in-time compilation to bytecode in some manner.

Try it out. Each link opens in a new tab.