When to use appendChild and when to use insertAdjacentHTML (and al alternative)

One thing that has always puzzled me is how to insert content into an existing document. In researching this I found two traditional alternatives and a newer API that may make things easier.

Recently I was playing with JSON-LD and trying to append content that used the JSON data into the page. The code looked like this:

```
const data =
  document.getElementById("data").text;
const json = JSON.parse(data);
const displayData = `
<h2>${json.name}</h2>
<h3>by ${json.author.name}</h3>
<p>${json.description}</p>
`;
```

The first idea was to use appendChild to append `displayData` to the document but it didn't work. `displayData` epxects a node as its parameter and `displayData` is not a node but a string.

The next option is to work with insertAdjacentHTML. it allows you to insert strings of HTML and to position them relative to a specific element. The positions of the inserted element are:

- `beforebegin`: Before the element itself.
- `afterbegin`: Just inside the element, before its first child.
- `beforeend`: Just inside the element, after its last child.
- `afterend`: After the element itself.

Note: `beforebegin` and `afterend` work only if the node is in the DOM tree and has a parent element.

`insertAdjacentElement` works fine but always having to specify the position of the element is prone to errors and there is no default.

The append method inserts a set of node or strings after the last child of the Element.

It combines the syntax of append`Child` with the flexibility of adding strings or DOM nodes, like we can with `insertAdjacentHTML`.

So, from research for this post, I find that append is the best method of the ones I researched to append content to the end of an existing element.

If the position of the data you’re inserting is important then `insertAdjacentHTML` is the method to use.