fetch

This package replaces the `http` package for HTTP calls. `fetch` package provides polyfill for the WHATWG fetch specification for legacy browsers or defaults to the global class which is available in modern browsers and Node. It is recomended that you use this package for compatibility with non-modern browsers.

For more information we recommend reading the MDN articles about it as this article covers only basic usage in Meteor.

Usage

Installation

To add this package to an existing app, run the following command from your app directory:

```
meteor add fetch
```

To add the `fetch` package to an existing package, include the statement `api.use('fetch');` in the `Package.onUse` callback in your `package.js` file:

```
Package.onUse((api) => {
  api.use('fetch');
});
```

API

You can import `fetch`, `Headers`, `Request` and `Response` classes from `meteor/fetch`.

```
import { fetch, Headers, Request, Response } from 'meteor/fetch';
```

For the most part though, you will only need to import `fetch` and `Headers`.

```
import { Meteor } from 'meteor/meteor';
import { fetch, Headers } from 'meteor/fetch';

async function postData (url, data) {
    const response = await fetch(url, {
        method: 'POST', // *GET, POST, PUT, DELETE, etc.
        mode: 'cors', // no-cors, *cors, same-origin
```

```
        cache: 'no-cache', // *default, no-cache, reload, force-cache, only-if-cached
        credentials: 'same-origin', // include, *same-origin, omit
        headers: new Headers({
            Authorization: 'Bearer my-secret-key',
            'Content-Type': 'application/json'
        }),
        redirect: 'follow', // manual, *follow, error
        referrerPolicy: 'no-referrer', // no-referrer, *no-referrer-when-downgrade, origin,
        body: JSON.stringify(data) // body data type must match "Content-Type" header
    });
    return response.json();
}

const postDataCall = Meteor.wrapAsync(postData);
const results = postDataCall('https://www.example.org/statsSubmission', { totalUsers: 55 })
```