The RestClient is a bit of a murky beast. Not perfectly documented for sure.

But it is in the end quite straightforward if you know how the whole thing works together.

1) Admin-On-Rest has defined some REST types (below). These are usually shot off by Redux actions (in their meta tag). The system scans for these rest types and if it sees them, then it calls the RestClient

GET\_LIST  
GET\_ONE CREATE  
UPDATE  
DELETE  
GET\_MANY GET\_MANY\_REFERENCE

The REST client is called with these types and some other params. It is the job of the rest client to interpret the type and then use the params to make a request to your API. For this AOR uses the new Fetch API that is built into browsers.

You can access it by calling. You should also go into AOR source code and check out how it works.

import { fetchUtils } from 'admin-on-rest';

2) The X total count is a header field that AOR needs for all responses to the GET\_LIST type. You can set this quite simply in your API. I use loopback and I set the X-Total-Count manually in a remote hook (don't worry about it if you don't know it) It seems your api is still using the JSON server. JSON server is a dummy API. So your app is not connected to your couchDB right now.

<https://github.com/typicode/json-server>

If you are not using an api server like express or loopback, then you can also configure your restClient do all request and response handling. You have to construct the URL. Read the below link so you can follow my example code further down.

<https://marmelab.com/admin-on-rest/RestClients.html#decorating-your-rest-client-example-of-file-upload>

so something like this.

if (type === 'GET\_LIST' && resource === 'posts') {

const url = http://127.0.0.1:5984/myproject/\_design/getclients/\_view/getclient/

options.method = 'GET';

return fetchUtils.fetchJson(url, options)

.then((response) => {

const {headers, json} = response;

//admin on rest needs the {data} key

return {data: json,

total: parseInt(headers.get('x-total-count').split('/').pop(), 10)}

})

You can also write a function like this to handle the request and response.

function handleRequestAndResponse(url, options={}) {

return fetchUtils.fetchJson(url, options)

.then((response) => {

const {headers, json} = response;

//admin on rest needs the {data} key

const data = {data: json}

if (headers.get('x-total-count')) {

data.total = parseInt(headers.get('x-total-count').split('/').pop(), 10)

} else {

data.total = json.length // this is why the X-Total-Count is needed by Aor

}

}

}

// handle get\_list responses

return {data: json,

total: } else {

return data

}

})

}

The above code has been formatted in the window and so might not work straight out of the box. But I hope you get the idea.