

Stocky

A Rest api interaction app, designed for fashion retailers

The purpose of this App was to improve on the product enumeration/importing and stock allocating aspects of the existing **EPOSNOW** backend.

As someone who has helped manage a chain of Retail shops, I found the generic product/stock adding facilities to be cumbersome. So I started laying out some requirements for a system that based on how I managed stock and inventory on spreadsheets for the shops I worked in previously.

In order to begin, I read through **EPOSNOW**'s developer documentation and saw that "backend" apps could be developed and integrated into their ecosystem. As the apps must run in the browser and run client side, **Javascript** was the logical choice.

EPOSNOW expose their database using a REST API endpoint, so I began writing a library that would encompass scheduled http requests (ajax), organised around specific API Endpoints (Brand, Colours, Products etc), all displayed with a **Bootstrap** GUI that thanks to **web workers** handling the inbound/outbound requests, is non-blocking.

Structure

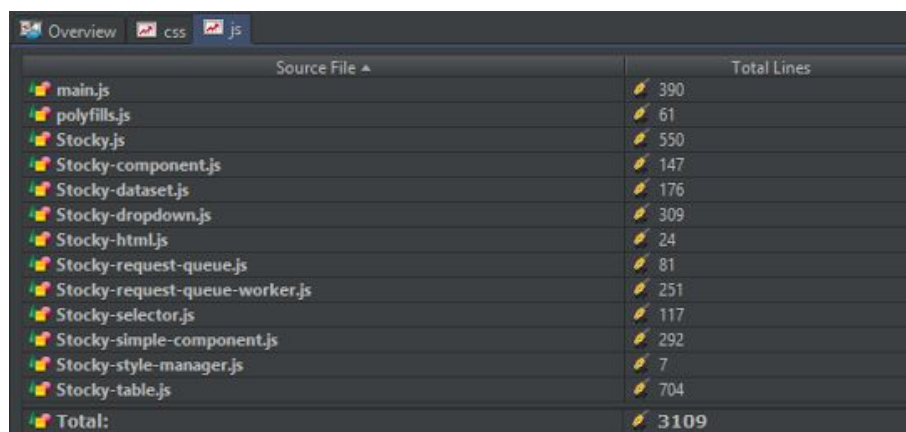
Object oriented design in javascript is tricky but not impossible, with module design patterns and function objects much of the Java-esk OO can be accomplished in **ECMA 5** (Browser compatibility). My design decisions are elucidated in the **Trello** (<https://trello.com/b/2zER6YS9/stocky-library>) which has my theoretical designs for the code but in reality the system as it stands in the current version works as a series of **Dataset** objects that all focus on a single endpoint.

Dataset

As explained in this card (<https://trello.com/c/ATWps8r4/12-worker-thread-threading-requests>) each endpoint such as the Brand endpoint works as a direct access to the Brand table in EPOSNOW's database. I concluded that it would make sense that any GUI Control that accessed the endpoint would want the most up to date version of that table. Each control then should be administered by a single dataset object that would handle all http requests, store an instance of the web workers, and update the data in all compositionally held controls if something has changed.

Size

Using Statistic in PHPStorm, these are the size values for the current build, along with all the class files laid out.



Source File	Total Lines
main.js	390
polyfills.js	61
Stocky.js	550
Stocky-component.js	147
Stocky-dataset.js	176
Stocky-dropdown.js	309
Stocky-html.js	24
Stocky-request-queue.js	81
Stocky-request-queue-worker.js	251
Stocky-selector.js	117
Stocky-simple-component.js	292
Stocky-style-manager.js	7
Stocky-table.js	704
Total:	3109

Ownership Evidence

Trello Board : <https://trello.com/b/2zER6YS9/stocky-library>

Eposnow Developer Profile:



Test Build On My Website: <http://stocky.dominicjomaa.com> (Current Test Bench So prone to breakage)

Git (Not in use currently Only For Backup) : <https://github.com/paperschool/Stocky>

Success (To be Created at spring term hand in)

Screenshots and Trello Explanations - (Sections not full layout)

Product Preset Editor (brands in this case), with validation working.

<https://trello.com/c/SM1Wu9w/9-interactive-table-object>

Brand Editor

Name

Ara

HB

Stonefly

Super Dry

Gucci

New Brand

Product Matrix creator - Creates potentially 100s of products by combining variants in a cartesian product calculation. <https://trello.com/c/kvrlAOoo/14-product-matrix>

Product Matrix Builder

Build Products Quickly and Easily

The product builder allows you to specify a products brand, colours and size range, then the product builder will create all the product combinations for you.

Product Style

Product Brand

Product Colours

Product Size-Range

Product Cost Price

Product Sale Price

Combo Dropdown Control - Designed for easy multi selection

Black, Suede, Orange, Maroon,

Black

Brown

Blue

Tan

Silver

Suede

Black Leather

Brown Leather

Orange

Mocca

Maroon

Charcoal

Mustard

Some Example debug from the HTTP Request threads.

<https://trello.com/c/ATWps8r4/12-worker-thread-threading-requests>

```
Running Initial Setup...
> Fetching App Settings...
{
  "THREAD": "LOCATION", "JOB TYPE": "GET", "JOB ID": 0, "ATTEMPTS": 0, "JOB OUTCOME": "SUCCESS", ...
  ATTEMPTS: 0
  JOB ID: 0
  JOB OUTCOME: "SUCCESS"
  JOB TYPE: "GET"
  THREAD: "LOCATION"
  URI: "https://api.eposnowhq.com/api/global/Location?page=1"
  __proto__: Object
}
{
  "THREAD": "COLOUR", "JOB TYPE": "GET", "JOB ID": 0, "ATTEMPTS": 0, "JOB OUTCOME": "SUCCESS", ...
}
{
  "THREAD": "CATEGORY", "JOB TYPE": "GET", "JOB ID": 0, "ATTEMPTS": 0, "JOB OUTCOME": "SUCCESS", ...
}
{
  "THREAD": "BRAND", "JOB TYPE": "GET", "JOB ID": 0, "ATTEMPTS": 0, "JOB OUTCOME": "SUCCESS", ...
}
{
  "THREAD": "PRODUCT", "JOB TYPE": "GET", "JOB ID": 0, "ATTEMPTS": 0, "JOB OUTCOME": "SUCCESS", ...
}
{
  "THREAD": "PRODUCT", "JOB TYPE": "GET", "JOB ID": 0, "ATTEMPTS": 0, "JOB OUTCOME": "SUCCESS", ...
}
{
  "THREAD": "COLOUR", "JOB TYPE": "GET", "JOB ID": 0, "ATTEMPTS": 0, "JOB OUTCOME": "SUCCESS", ...
}
{
  "THREAD": "LOCATION", "JOB TYPE": "GET", "JOB ID": 0, "ATTEMPTS": 0, "JOB OUTCOME": "SUCCESS", ...
}
{
  "THREAD": "BRAND", "JOB TYPE": "GET", "JOB ID": 0, "ATTEMPTS": 0, "JOB OUTCOME": "SUCCESS", ...
}
{
  "THREAD": "CATEGORY", "JOB TYPE": "GET", "JOB ID": 0, "ATTEMPTS": 0, "JOB OUTCOME": "SUCCESS", ...
}
{
  "THREAD": "PRODUCT", "JOB TYPE": "GET", "JOB ID": 0, "ATTEMPTS": 0, "JOB OUTCOME": "SUCCESS", ...
}
{
  "THREAD": "PRODUCT", "JOB TYPE": "GET", "JOB ID": 0, "ATTEMPTS": 0, "JOB OUTCOME": "SUCCESS", ...
}
```