Webalacrity Op 1 No 1
Programmer's Journal

Recipe Manager

Programming Tools (Windows Implementation)

- Angular (v. 14.0.3) - installed globally with NPM
- NodeJS (v. 16.15.1) - backend JavaScript environment
- Elasticsearch (8.3.1) - backend database + NoSQL query language
- Kibana (8.3.1) - Elasticsearch data viewer and manager
- NPM Module Extras
  - Bootstrap (5.2.0)
  - jQuery (3.6.0)
  - Express (4.18.1)
  - Elasticsearch Client (8.2.1)

Prerequisites

- JavaScript proficiency
- Proficiency with a source code editor/compiler (e.g. Visual Studio Code)
- JSON and JavaScript Object format familiarity
- Some acquaintance with Angular
- Basic knowledge of Visual Studio Code editor
- Concepts of source control
- Basic knowledge of NPM
- Basic knowledge of what an API is
- Browser debugger familiarity
- Basic knowledge of GitHub and Git

Tool Installations

- NPM via NodeJS
- Visual Studio Code
- Git Bash
- Git for Windows
- Angular
- Angular Material
- Bootstrap
- jQuery
- Elasticsearch
- Kibana

NodeJS

The installation of NodeJS will automatically install NPM. NPM is used to install NodeJS modules within Angular and NodeJS backend.

Installation Website

NodeJS

Visual Studio Code

Download:

Visual Studio Code

This is a wonderful source editor and compiler for many different programming languages. One can use this to debug and step through code on the backend or frontend.

Git Downloads

Git Website

Git is the source control tool to keep track of all programming changes. The host used here is GitHub. Very useful in creating branches to source, checking out previous versions and adding tags to the Git source tree.

All the code for this project will be available online at GitHub which can be downloaded using git commands.

GitHub

Location for source code changes.

- Initialize project with a git repository.
- Initial commit of start of project.
- Link project to GitHub so changes can be push up to offsite repo.
- Clone GitHub repo to download source code locally.

Angular Global Install (NPM)

Install Angular globally using NPM.

Command line:

```
npm uninstall -g @angular/cli
npm install -g @angular/cli@latest
```

Webalacrity Op 1 No 2
Programmer's Journal

Recipe Manager

Initialize a new Angular Project

Create a new Recipes project

```
ng new recipes
```

Angular Material (angular cli)

Installation within Angular CLI

Used for GUI enhancements, e.g. Drag and Drop interface.

```
ng add @angular/material
```

Bootstrap (NPM)

Installation:

```
npm i bootstrap@5.2.0-beta1 --save
```

```
npm i bootstrap --save
```
(to get the latest stable)

Front-end toolkit which can be used in Angular by installing the module using NPM. Provides very attractive user interface widgets.

Add to Angular.json configuration

```
"styles": [
            "node_modules/bootstrap/dist/css/bootstrap.min.css",
              "src/styles.css"
],
```

jQuery (NPM)

Installation:

```
npm i jquery --save
```

Add to Angular.json configuration

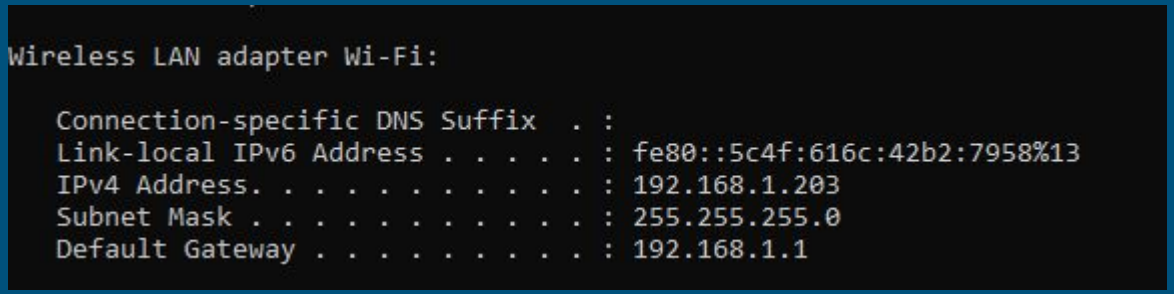
```
"scripts": [
"node_modules/jquery/dist/jquery.min.js"
]
```

Used for accessing some legacy functionality provided by JavaScript jQuery. Typically accessing the jQuery object is via the $ symbol.

Checking the IP address of Computer

From the command line

```
ipconfig
```



The IPv4 Address gives you the IP address we will use to create a local server to handle security and access to database. It is also the address to use to configure elasticsearch and kibana. In this case IP = 192.168.1.203

Webalacrity Op 1 No 3
Programmer's Journal

Recipe Manager

Elasticsearch & Elasticsearch Client (NPM)

Download site:

Elasticsearch (backend installation from zip file)

Installation into Angular via NPM:

```
npm install @elastic/elasticsearch --save
```

Used at the backend to access Elasticsearch database using queries.

Kibana

Download site:

Kibana (backend installation from zip file)

An excellent online Elasticsearch manager of all the documents and indices stored in Elasticsearch. Can do queries on-the-fly. Allows you to easily create new indices of documents and experiment with data.

Setup Kibana security from Elasticsearch setup

Security Setup for Kibana

```
kibana-setup --enrollment-token <token>
```

The token can be found after the output of the first execution of elasticsearch program from command line. You find the token in this line of the elasticsearch terminal output. This is generated randomly in elasticsearch.

```
->  Configure Kibana to use this cluster:

* Run Kibana and click the configuration link in the terminal when Kibana starts.

* Copy the following enrollment token and paste it into Kibana in your browser (valid for the
next 30 minutes):
```


```
eyJ2ZXIiOiI4LjMuMCIsImFkciI6WyIxOTIuMTY4LjEuMjAzOjkyMDAiXSwiZmdyIjoiNWEwOTM0MjkwNmFjYzI0MzM1MjI
2ZTNkNzEzZDRiYzRjZDQ4ZGVmODBmZjBiMWYxZmE4OWFkZDI0NGRhODBkOSIsImtleSI6IjJsVVBzSUVCR1p4RXNMemlCcU
hvOjdTejVLbW1rUm1TRzYwV2JWNUZyMUEifQ==
```

First Launch of Kibana

The first time you launch Kibana, you will required put enter credentials.

User Name: elastic

PW: QvCfZ4hqE7DJ+EcKQj4e (randomly output from elasticsearch message on first launch)

Elasticsearch Cluster

See this location of definitions of clusters, nodes, replicas, shards, etc.

Elasticsearch Cluster

- Cluster = 3 or more nodes
- Node: is a single ES instance
- Index: collection of ES documents distributed among nodes
- Document: analogous to a record instance in a relational database
- Shard: index made up of shards each containing parts of index. Each index contains one primary shard and one replica
- Replica: used to back-up documents
- All data is defined in JSON format in Elasticsearch
- Elasticsearch is a NoSQL database and uses a special query language to access the data (also defined in JSON format)

Checking the status of Elasticsearch Shards

https://192.168.1.203:9200/_cluster/health?pretty

https://192.168.1.203:9200/_cat/shards

In order to move the status from yellow to green you need replicas of the data which means the database is clustered on different computers (or nodes).

The IP address 192.168.1.203 is defined on your particular computer and may differ from the above example. Use ipconfig in DOS command window to find your address.

JSON

JSON validator:

JF Best JSON Formatter

JSON is used as the primary data format for configuration and data storage.
JSON is the text format for JavaScript Objects.

Webalacrity Op 1 No 4
Programmer's Journal

Recipe Manager

Initializing Git Repo for Project

Navigate to your project folder and from the command line invoke the following command. This will create a hidden .git folder in your project folder and will represent your code changes repository. After initialization open the Git-GUI program and open your project folder to see your repository status.

```
git init
```

Recipe Manager Web App Features

- Recipe Database and Menu
- Recipe Editor
- In Stock Inventory Tracker
- Shopping List based on menu selection
- Drag-and-Drop shopping list using inventory items
- Accessory List
- Recipe Search
- Front-end using Angular
- Back-end using Elasticsearch as database and query
- REST API interface to data
- Accessible from mobile devices such as phones and iPAD
- Accessible on home WiFi using IP of computer
- Potentially accessible from a public IP address and domain URL
- Export data to JSON files for backup
- Import data from JSON files to initialize database

Defining the Data Model - Recipe

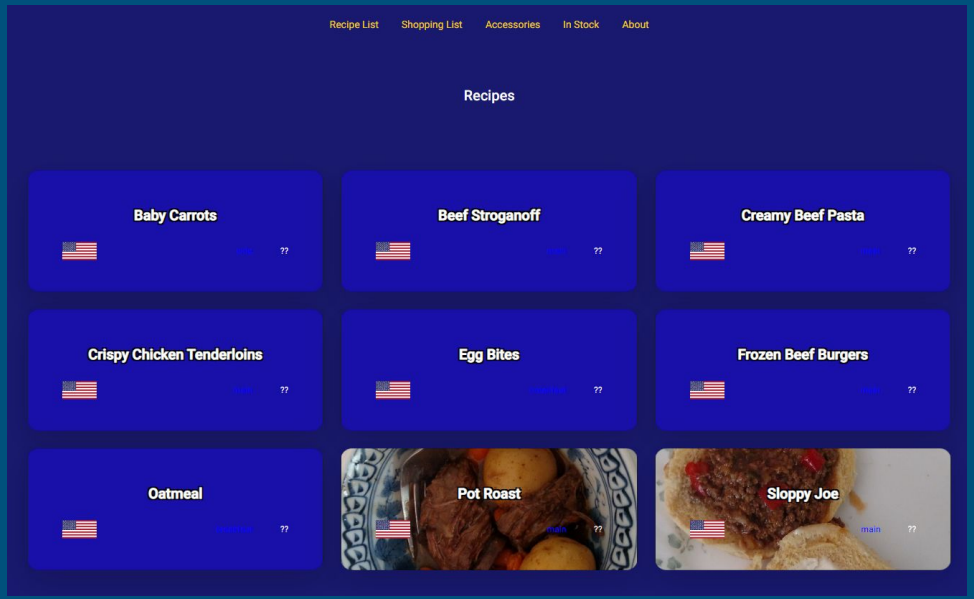
- identifier (keyword)
- name (text)
- mealType (text)
- linkName (text)
- notes (text)
- ingredients (array)
  - Item (number)
  - description (text)
- instructions (array)
  - step (number)
  - amount (text)

Defining the Data Model - Inventory

- identifier (keyword)
- category (text)
- name (text)
- instock (boolean)

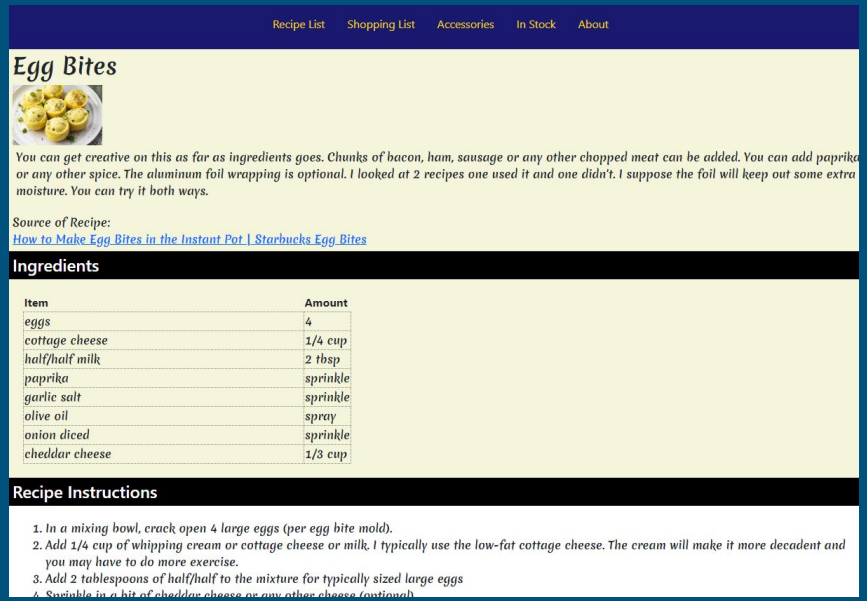
Design Concept

Recipe List Choices (landing page):



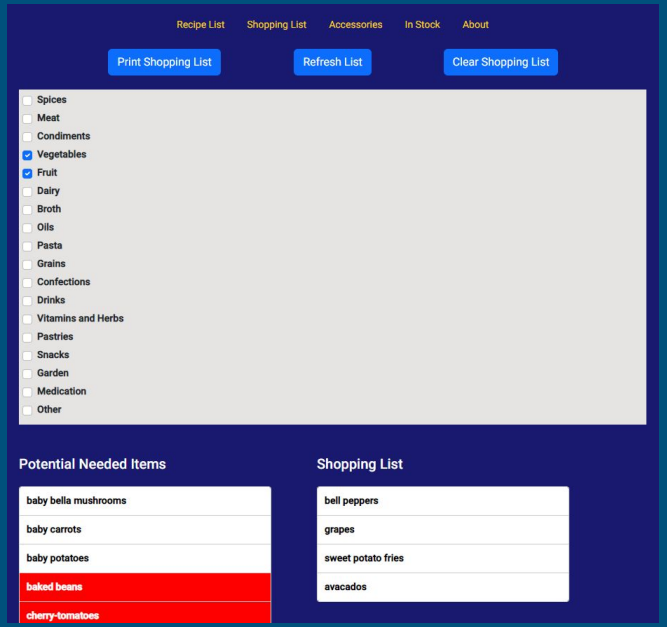
Design Concept

Recipe Detail Page:



Design Concept

Shopping List Page (drag and drop)



Design Concept

Inventory Checklist Page(in/out of stock items)

Design Concept

Accessories & Conversions Page

Recipe List Shopping List Accessories In Stock About

Accessories

Contained: Set of glass containers. Bought at King Soopers.

- Microwaveable but will get hot
- Oven safe up to 450 degrees
- Dishwasher safe
- Freezer Safe

Silicon Sling for the Instant Pot (Walmart)

Silicon Egg Bite Mold for the Instant Pot (Walmart)

Green Bowl which weighs 6.5 oz - used for weighing ingredients

Digital Kitchen Scale (Dash of That)

Ingredient Brand Names

- Aiello's Spaghetti Sauce
- Barilla Mini Farfalle Bowtie Pasta (1 lb)
- Blue Elephant Royal Thai Cuisine Coconut Cream (Sprouts)
- Diced Mirepoix Mix (Onions, Carrots, Celery) (Walmart)
- Gold Meadow Viva Low-fat Cottage Cheese (Walmart)
- Goodcook Touch Veggie Dicer
- Kroger Non-stick Extra Virgin Olive Oil Cooking Spray
- Kroger Garlic Minced
- Kroger Garlic Salt
- Kroger Italian Seasoning
- Kroger Sliced Mushrooms Baby Bella
- Kroger Marinara Pasta Sauce (24 oz)
- Ore-Ida Sweet Frozen Potato Crinkle Cut Fries (French Fried Sweet Potatoes)
- Pacific Foods Organic Beef Broth
- Pompeian Extra Virgin Olive Oil
- Quaker Oats Old Fashioned (King Soopers or Walmart)
- Simple Truth Frozen Sausage Links (King Soopers)

Webalacrity Op 1 No 5
Programmer's Journal

Recipe Manager

Recipe Index Schema (settings)

```
PUT recipe
{
  "settings": {
    "analysis": {
      "analyzer": {
        "comma_delimted_analyzer" : {
          "type": "custom",
          "tokenizer": "comma_delimted_tokenizer" ,
          "filter": [
            "lowercase",
            "asciifolding"
          ]
        }
      },
      "tokenizer": {
        "comma_delimted_tokenizer" : {
          "type": "pattern",
          "pattern": "(, *)"
        }
      }
    }
  }
}
```

Recipe Index Schema (mapping)

```
PUT recipe/_mapping
{
  "properties" : {
    "identifier" : {
      "type" : "text" ,
      "analyzer" : "keyword"
    },
    "name" : {
      "type" : "text" ,
      "analyzer" : "keyword" ,
      "fields" : {
        "standard" : {
          "type" : "text" ,
          "analyzer" : "standard"
        }
      }
    },
    "mealType" : {
      "type" : "text" ,
      "analyzer" : "keyword"
    },
    "linkName" : {
      "type" : "text" ,
      "analyzer" : "english"
    },
    "linkSource" : {
      "type" : "text" ,
      "analyzer" : "keyword"
    },
    "notes" : {
      "type" : "text" ,
      "analyzer" : "english"
    },
    "ingredients" : {
      "properties" : {
        "item" : {
          "type" : "text" ,
          "analyzer" : "keyword" ,
          "fields" : {
            "standard" : {
              "type" : "text" ,
              "analyzer" : "standard"
            }
          }
        },
        "amount" : {
          "type" : "text" ,
          "analyzer" : "english"
        }
      }
    },
    "instructions" : {
      "properties" : {
        "step" : {
          "type" : "long"
        },
        "description" : {
          "type" : "text" ,
          "analyzer" : "english"
        }
      }
    }
  }
}
```

Webalacrity Op 1 No 6
Programmer's Journal

Recipe Manager

Adding a new repo to GitHub

```
git remote add origin <your github account>/<project name>.git
```

```
git push -u origin master
```

This will store some information of the GitHub site into your .git/config file automatically and tie your local repo to the GitHub site. Not needed by user unless you are creating your own GitHub project. Once done, you can push your committed changes to GitHub or allow others to clone your project.

Cloning a project from GitHub

In a Windows command line, go to a directory you want to contain a GitHub project. Use this command to clone the project files.

```
git clone <project URL>
```

To clone this tutorial's source code project and slides:

```
git clone https://github.com/JSegor/jeff_recipes_tutorial
```

Navigate into the recipes folder and do the following command to restore the node_modules directory.

```
npm install
```

NPM will look at the package.json file to restore necessary packages. You can then run the proxy server using

```
ng serve
```

Go to a browser and enter

```
localhost:4200
```

to bring up our blank website.

Webalacrity Op 1 No 7
Programmer's Journal

Recipe Manager

Turning your angular app into a server

In order for your web application to be available on other devices on your WiFi network you will need to turn it into a server. This will create a true NodeJS backend to communicate to using REST API commands that you define. These commands will allow data to be retrieved from the Elasticsearch database publicly on the network. You need to install the NPM package in your project:

```
npm install express --save
```

REST API Definitions

These sites will give you a good introduction into what is a REST API. We will be using this architectural style when communicating between Angular client and the NodeJS backend. The data flow will look like this:

Angular Client (TypeScipt) ← → Restful API ← → Server (NodeJS) ← → Restful API ← → Elasticsearch (Java)

REST Architectural Constraints

What is REST?

Elasticsearch uses a special JSON-based query language to access database. Express makes it easier for the client to talk with server and Elasticsearch Client makes it easier for server to talk with Elasticsearch nodes (processes). The Angular HttpClient assists in the Restful API between client and server.

Creating a real NodeJS service call

In order to create a backend server (which has access to database) for this project you will need to create a NodeJS service which you can call from the command line. All URL calls to your app which are not caught by Angular will be routed to the Backend server code. We will no longer be using `ng serve` to debug after this. The essence of the server code (server.js) to do this is:

```
app.use('/api', api);
app.use('/api-import', apiImport);

// Catch all routes
app.get('*', (req, res) => {
    res.sendFile(path.join(__dirname, 'dist/recipes/index.html' ))
});

const port = process.env.PORT || 98;

const ip = process.env.IP || '192.168.1.203';

app.listen(port, ip, (req, res) => {
    console.log(`Running on ip: ${ip} port: ${port}`);
});
```

NodeJS backend code

The actual NodeJS backend code which will intercept your REST API calls will have the following form:

```
'use strict'

const express = require('express');
const fs = require('fs');
const router = express.Router();

// These two statements allow parsing of body correctly when
// getting a POST request (otherwise req.body may equal undefined)
router.use(express.json());
router.use(express.urlencoded({ extended: true }));

// Get all recipes
router.get('/recipes/all-recipes', (req, res) => {
  // code to access database goes here. This intercepts a specific REST API call to retrieve all recipes and return to client
)};

module.exports = router;
```

Webalacrity Op 1 No 8
Programmer's Journal

Recipe Manager

Using Template Literals

For convenience, instead of hard-coding the http://192.168.1.203:98 address within the code we can use template literals and pull the IP and Port parts from environment variables. For example:

```
`https://${process.env.SERVER_IP}:${process.env.ES_PORT }`
```

Or

```
const port = process.env.SERVER_PORT || 4200;

const ip = process.env.SERVER_IP || 'localhost';
```

So if we have to change the IP/Port address, we only have to change it in the environment variables, not the code.

Changing your Windows Environment Variables

If you use Windows' search box for finding the environment variables window you may not be able to change your system environment variables (read-only). If this is the case, bring up the environment variables by:

1. Win Key + X key
2. Type Y
3. Click 'Advanced System Settings'
4. Choose 'Environment Variables'
5. Add your new system variables (lower window, 'new' button)

Add following Windows system variables

ES_USER_ID = elastic

ES_USER_KEY = <*password you received on setup of elasticsearch*>

ES_PORT = 9200

SERVER_IP = <*IP address of your computer from ipconfig. e.g 192.168.1.203*>

SERVER_PORT = 98 (*an arbitrary free port*)

These will be accessed by the NodeJS code.

Listing environment variables from PowerShell

To bring up a PowerShell window:

1) Win Key + X
2) A-key
3) `gci env:SERVER*`
4) `gci env:ES*`

Create new angular components

```
ng g c landing-page --skip-tests

ng g c navigator --skip-tests

Ng g c import-data --skip-tests
```

These the LandingPageComponent will be the home page. The NavigatorComponent will be the top navigator menu to other pages. The ImportDataComponent will be navigated to and will have buttons to import recipes and instock items.

Modify Routings

`App-routing.module.ts`

Modify this file to add default home page to our landing page.

```
const routes: Routes = [
  {
    path: '',
    redirectTo: '/landing',
    pathMatch: 'full'

  },
  {
    path: 'landing', component: LandingPageComponent
  },
```

Insert the Navigation element

The navigation component is placed in the app.component.html file:

```
<app-navigator></app-navigator>

<router-outlet></router-outlet>
```

The router-outlet is the location where components are placed when switching in and out of components based on URL changes in the routing. The navigator component is placed here because we want it to be the header of all other pages we are switching to.

Webalacrity Op 1 No 9
Programmer's Journal

Recipe Manager

Finish adding remaining components

```
ng g c accessories --skip-tests

ng g c error-page --skip-tests

ng g c in-stock --skip-tests

ng g c recipe-holder --skip-tests

ng g c shopping-list --skip-tests

ng g s data-access --skip-tests (service)
```

Using the Visual Studio Code debugger

We can step through code both in the front end (Angular TypeScript) or back end (NodeJS) using a different way of starting the application. Use the launch.json file to configure these two launchers.

Debugging Angular - Launch Chrome (two output windows: terminal & debug console)

Debugging NodeJS - Launch Program (one output window: debug console)

When launching from Chrome, the terminal window is output from NodeJS server, and the debug console is output from Angular front-end.

When launching from Progam (NodeJS) only the debug console is used for output from NodeJS server.

Launch Program Needs Working Directory

```
{
        "name": "Launch Program",
        "program": "${workspaceFolder}recipes/server.js",
        "request": "launch",
        "skipFiles": [
            "<node_internals>/**"
        ],
        "type": "node",
        "cwd":"${workspaceFolder}/recipes"
}
```

Need to configure angular.json for debug

In order to step through Angular TypeScript code you need to switch from the “production” default configuration to “development” where it sets the sourceMap = true. Otherwise, if you set breakpoints in code and you run the app, the breakpoints will become unbound. By default, the default configuration is set to “production”.

Production mode also makes the source code in the Chrome debugger unreadable and can’t use its debugger.

Release the application in production mode, however.

To avoid unbound breakpoints when debugging

1) Make sure defaultConfiguration is set to “development” in angular.json.
2) Make sure the launch.json configuration for Chrome has the correct working folder path, so include the recipes folder:

```
"configurations": [
 {
     "name": "Launch Chrome on Server",
     "request": "launch",
     "type": "chrome",
     "url": "http://192.168.1.203:98",
     "webRoot": "${workspaceFolder}/recipes"
 }
```

Use this Kibana Command to remove documents

To remove all documents from the recipe index without removing the index itself, use the following command in Kibana. Do the same for the inventory index if you need to start from scratch with a new import of data.

```
POST recipe/_delete_by_query?conflicts=proceed

{

  "query": {

    "match_all": {}

  }

}
```

Angular Property Binding & String Interpolation

The following types of data binding is crucial in Angular apps:

- Event binding
- Two-way binding
- String Interpolation
- Property Binding
- Attribute Binding

See this source for details:

Understanding Angular Property Binding and Interpolation

Note: Attributes are characteristic of HTML and Properties are characteristic of the DOM. The names could be different. Property Binding binds to the properties of the DOM.