Convert MVC Drop Down Lists and Perform Searching Using Angular

Paul D. Sheriff
PRESIDENT, PDSA, INC.

@pdsainc www.pdsa.com psheriff@pdsa.com

Goals

Build drop down list

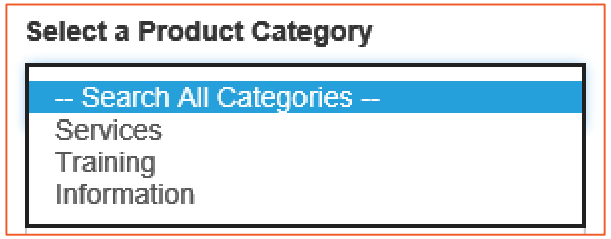
Build more Web API calls

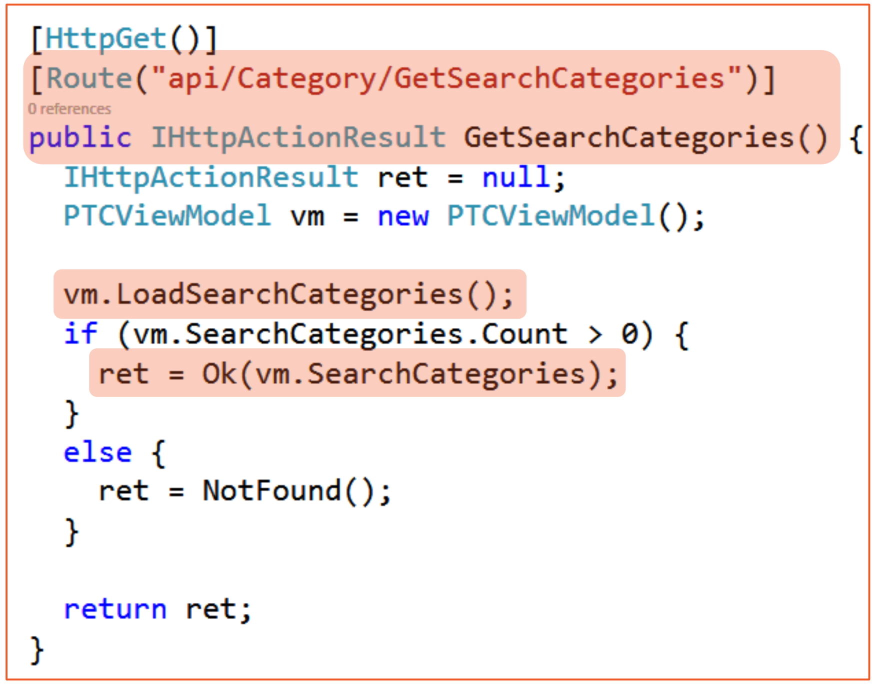
Handle button click events

Bind input fields to controller properties

Multiple ways to search

Build a Drop Down List





```
[HttpGet()]
[Route("api/Category/GetSearchCategories")]
0 references
public IHttpActionResult GetSearchCategories() {
  IHttpActionResult ret = null;
  PTCViewModel vm = new PTCViewModel();

  vm.LoadSearchCategories();
  if (vm.SearchCategories.Count > 0) {
    ret = Ok(vm.SearchCategories);
  }
  else {
    ret = NotFound();
  }

  return ret;
}
```

Add new Web API controller

- CategoryController

Build new Web API call

- GetSearchCategories()
- Use Route attribute

Load SearchCategories collection in view model

Serialize collection as JSON

- Return using Ok()

```
vm.searchCategories = [];
vm.searchInput = {
  selectedCategory: {
    CategoryId: 0,
    CategoryName: ''
  },
  productName: ''
};
```

```
function searchCategoryList() {
  dataService.get("/api/Category/GetSearchCategories")
   .then(function (result) {
     vm.searchCategories = result.data;

   }, function (error) {
     handleException(error);
   });
}
```

Create two new properties on scope

- searchCategories array
- searchInput object

Create searchCategoryList() function

Load categories using data service

Store data into new searchCategories array

```
vm.searchCategories = [];
vm.searchInput = {
  selectedCategory: {
    CategoryId: 0,
    CategoryName: ''
  },
  productName: ''
};
```

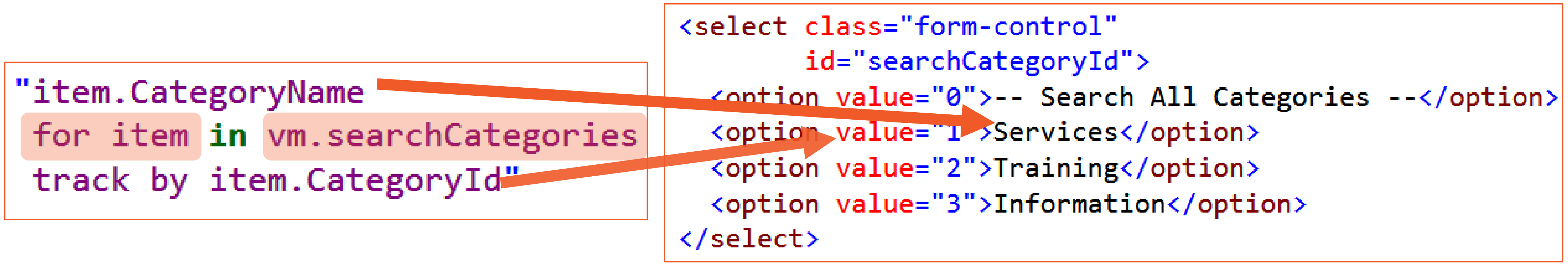
```
<label for="searchCategoryId">
  Select a Product Category
</label>
<select id="searchCategoryId"
        class="form-control"
        ng-model="vm.searchInput.selectedCategory"
        ng-options="item.CategoryName
                    for item in vm.searchCategories
                    track by item.CategoryId">
</select>
```

Angular directives for <select> element

- A little more involved than a table

Two directives

- ng-model = bind to selectedCategory
- ng-options = defines how to load the <option> elements



```
"item.CategoryName
 for item in vm.searchCategories
 track by item.CategoryId"
```

```
<select class="form-control"
        id="searchCategoryId">
  <option value="0">-- Search All Categories --</option>
  <option value="1">Services</option>
  <option value="2">Training</option>
  <option value="3">Information</option>
</select>
```

Item.CategoryName = text for <option>

- <option>Services</option>

"item" = local variable

"vm.categories" = collection in scope

track by "item.CategoryId" = id to put into value of <option>

- <option value="1">Services</option>

Demo

Category drop down

Event Handling and Data Binding

```
vm.searchCategories = [];
vm.searchInput = {
  selectedCategory: {
    CategoryId: 0,
    CategoryName: ''
  },
  productName: ''
};
```

```
<label for="searchProductName">
  Search for Product Name
</label>
<input id="searchProductName"
       ng-model="vm.searchInput.productName"
       type="text"
       class="form-control" />
```

Replace Razor product name search field with HTML

Bind searchInput.productName

```
function resetSearch() {
  // Clear search entries
  vm.searchInput = {
    selectedCategory: {
      CategoryId: 0,
      CategoryName: ''
    },
    productName: ''
  };

  // Requery to get any new data
  productList();
}
```

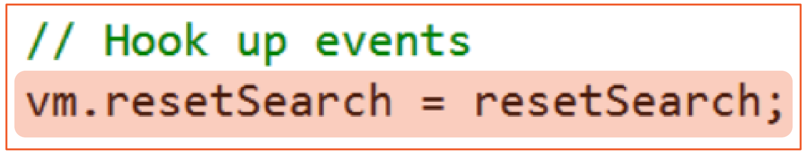
Create resetSearch() function

Set vm.searchInput back to blank values

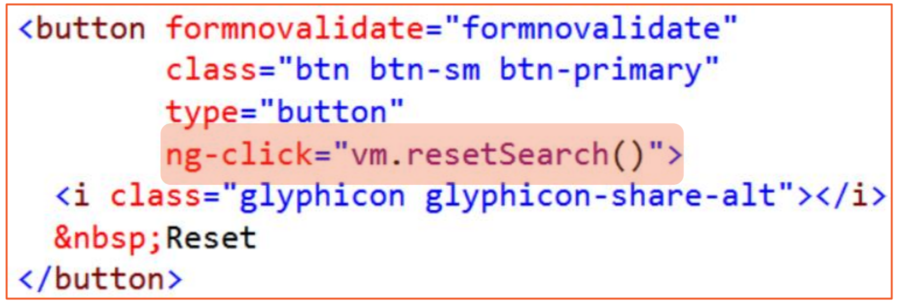
- Automatically updates bound fields

Optionally call productList()

- In case other data has been added



```javascript
// Hook up events
vm.resetSearch = resetSearch;
```



```html
<button formnovalidate="formnovalidate"
        class="btn btn-sm btn-primary"
        type="button"
        ng-click="vm.resetSearch()">
  <i class="glyphicon glyphicon-share-alt"></i>
  &nbsp;Reset
</button>
```

Call resetSearch from “Reset” button

Add resetSearch to scope

- Allows function to be called from HTML

Use ng-click to call function

- Don’t use onclick

Demo

Events and data binding

Built-In Searching

Client-Side Searching with Angular

Search all properties in object

Search a single property

Write function to search

```
<tr ng-repeat="product in vm.products
    | filter: vm.searchInput.productName">
```

Add “filter” in ng-repeat directive

Bound controller variable on input field

- vm.SearchInput.productName

Performs “like” search across all properties in product object

```
<tr ng-repeat="product in vm.products
         | filter: {ProductName:vm.searchInput.productName}">
```

Search a single property

Format {propertyName:controllerVariable}

- Property name in object to search
- Bound controller variable on input field

```
<tr ng-repeat="product in vm.products
    | filter: vm.searchImmediate ">
```

Search on multiple properties

Complicated search

Specify function on scope to search

Write code to search any way you want

Demo

Built-in searching

Search Via Web API

Server-Side Searching

Want to get new data

Create new Web API for searching

Build object literal from input

Post object literal to Web API

Get new data from database

Send back to Angular

Demo

Search via Web API

Summary

Built category <select> list

Bind searching input fields

Hook up events

Multiple ways to search

Coming up in the next module…

- Show/hide HTML elements
- Keep track of page “state”
- Handle exceptions