STAYING SHARP with Angular.js

Directives Level 2

Scope the Config Object Section 2

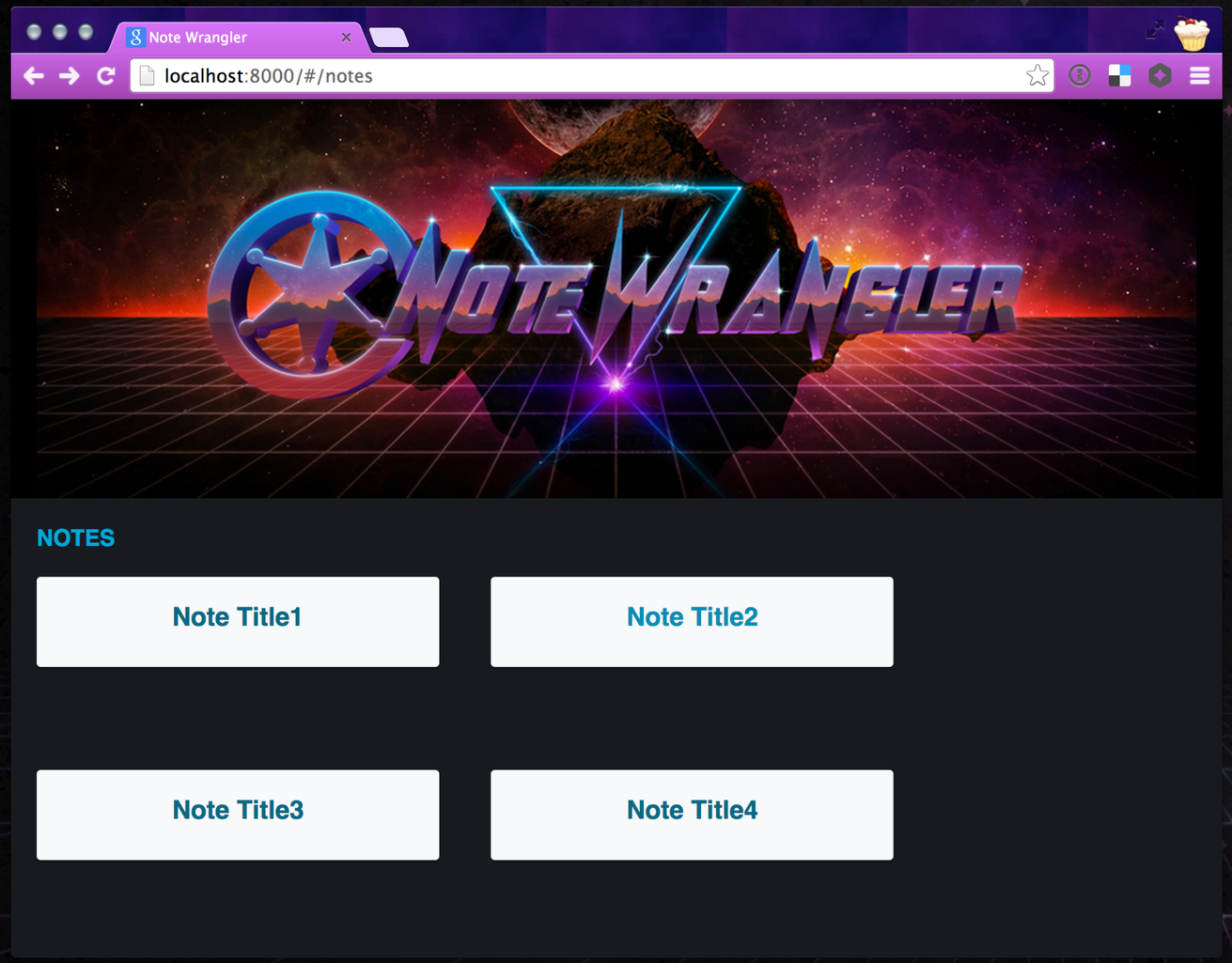
STAYING SHARP with Angular.JS

Current Code

nw-card.js

```
angular.module("NoteWrangler")
.directive("nwCard", function() {
 return {
    restrict: "E",
    templateUrl: "templates/directives/nw-card.html",
    controller: function($scope){
     $scope.header = "Note Title";
    }
 };
});
```

We Want to Number the Notes



Understanding $scope

We are going to simplify our note example and add numbers to the headers.

notes.html

```
...
<div class="note-wrapper">
  <a class="card-notes">
    <nw-card></nw-card>
    <nw-card></nw-card>
    <nw-card></nw-card>
    <nw-card></nw-card>
  </a>
</div>
```

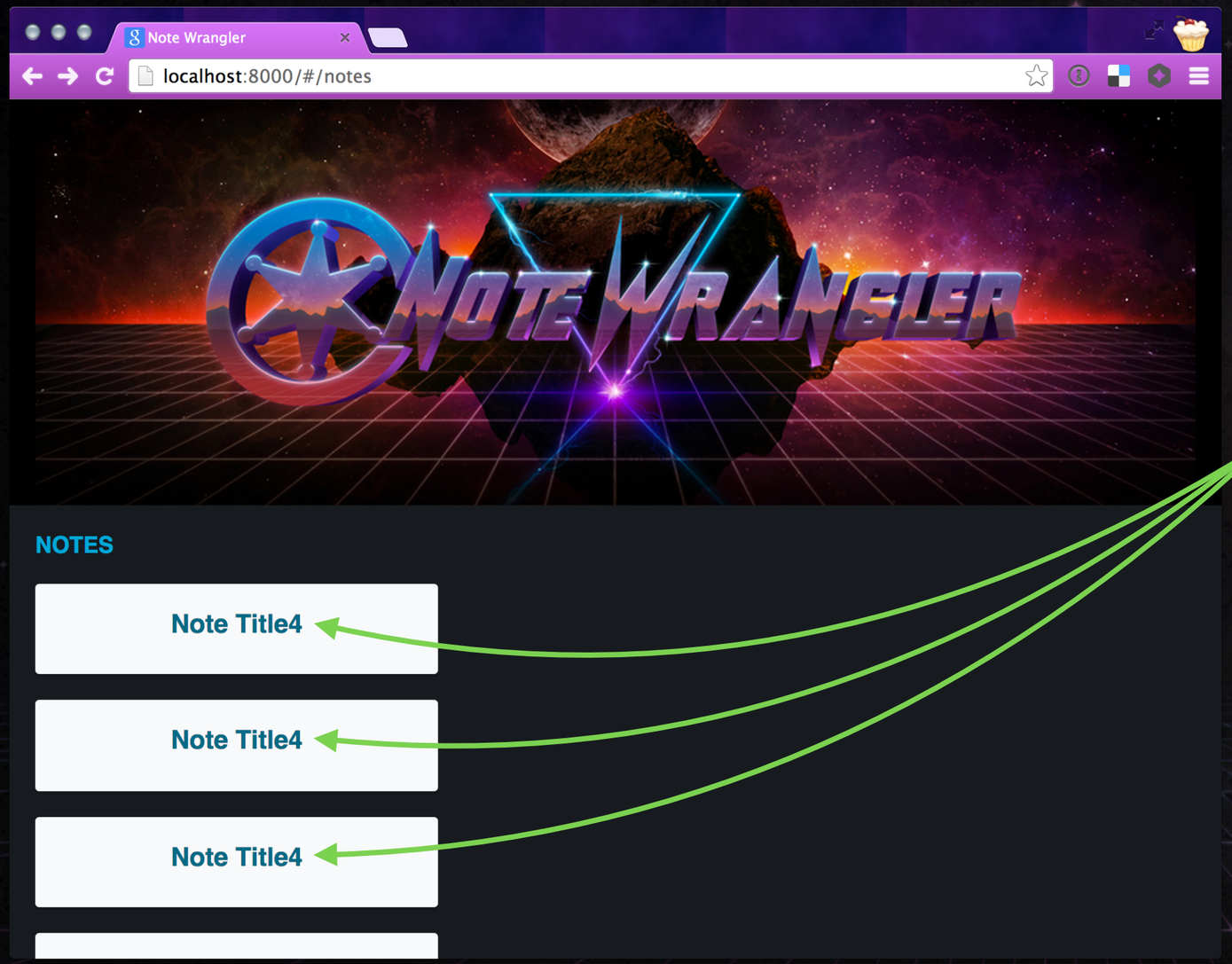
example/nw-card.html

```
<div class="card">
  <h2 class="h3">{{header}}</h2>
</div>
```

nw-card.js

```
angular.module("NoteWrangler")
.directive("nwCard", function() {
  var num = 1;
  return {
    restrict: "E",
    templateUrl: "example/nw-card.html",
    controller: function($scope){
      $scope.header = "Note Title" + num++;
  };
});
```

Inherited Scope Can Be a Problem



All the same number?!

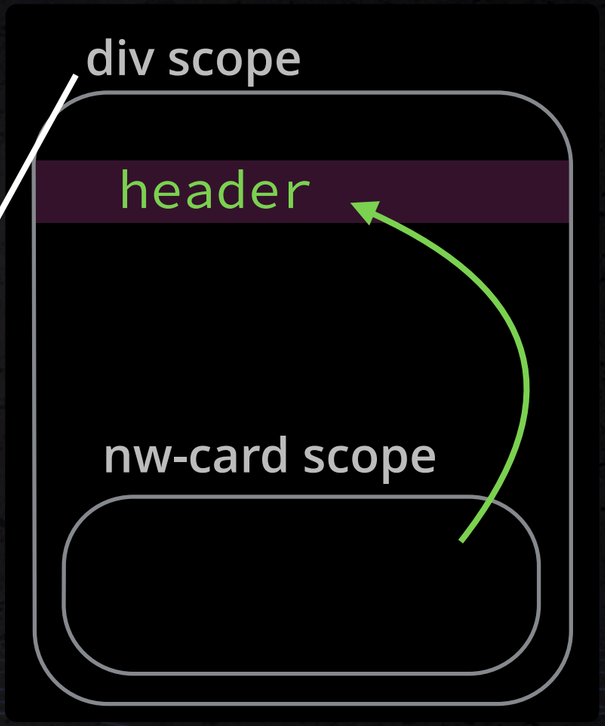
Directives Inherit Their Parent's Scope by Default

Our parent is the next directive up in the hierarchy that creates a new scope.

index.html

```
<html lang="en" ng-app="NoteWrangler">
  <head>...</head>
  <body>
    <div class="nav-list">
      <a href="#/notes"> Notes </a>
      <a href="#/users"> Users </a>
    </div>

    <div class="hero-wrapper"> ... </div>
    <div class="main-wrapper">
      <div ng-view></div>
    </div>
    <script src="/js/vendor/jquery.js"></script>
```

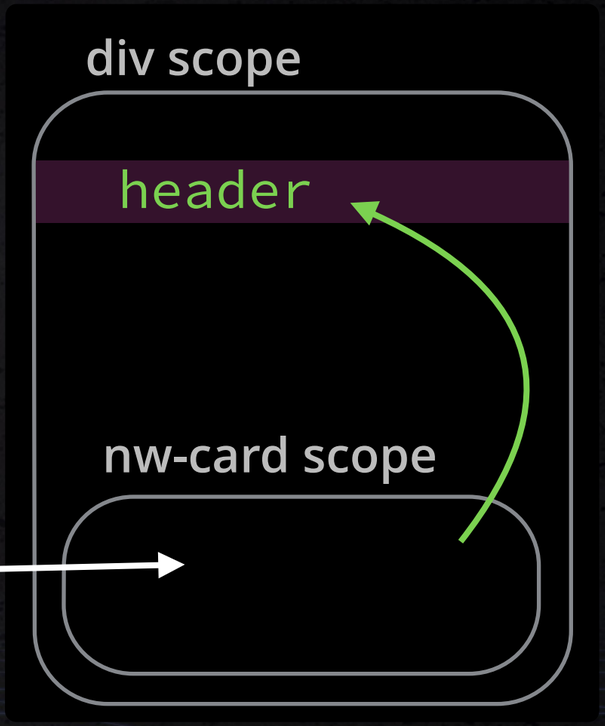


Directives Inherit Their Parent's Scope by Default

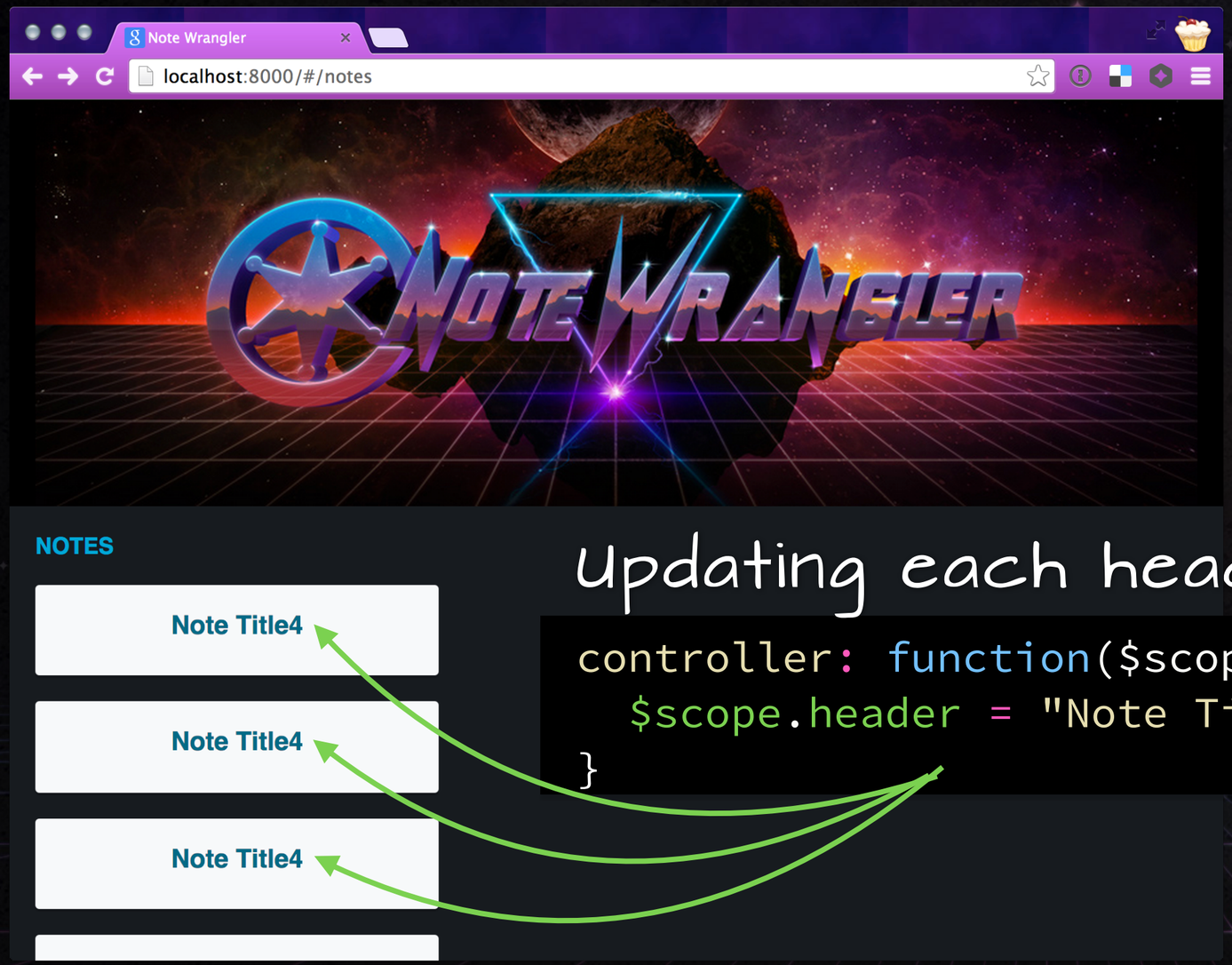
This wouldn't work how we want it to, though, because of inherited scope.

nw-card.js

```
angular.module("NoteWrangler")
.directive("nwCard", function() {
  var num = 1;
  return {
    restrict: "E",
    templateUrl: "templates/directives/nw-card.html",
    controller: function($scope){
      $scope.header = "Note Title" + num++;
    }
...
```



Inherited Scope Can Be a Problem



Updating each header every time

```
controller: function($scope){
  $scope.header = "Note Title" + num++;
}
```

Isolating the Scope

By passing an object to the scope option, you are creating an isolate scope. This tells the directive to keep scope inside of itself and not to inherit or share with other scopes.

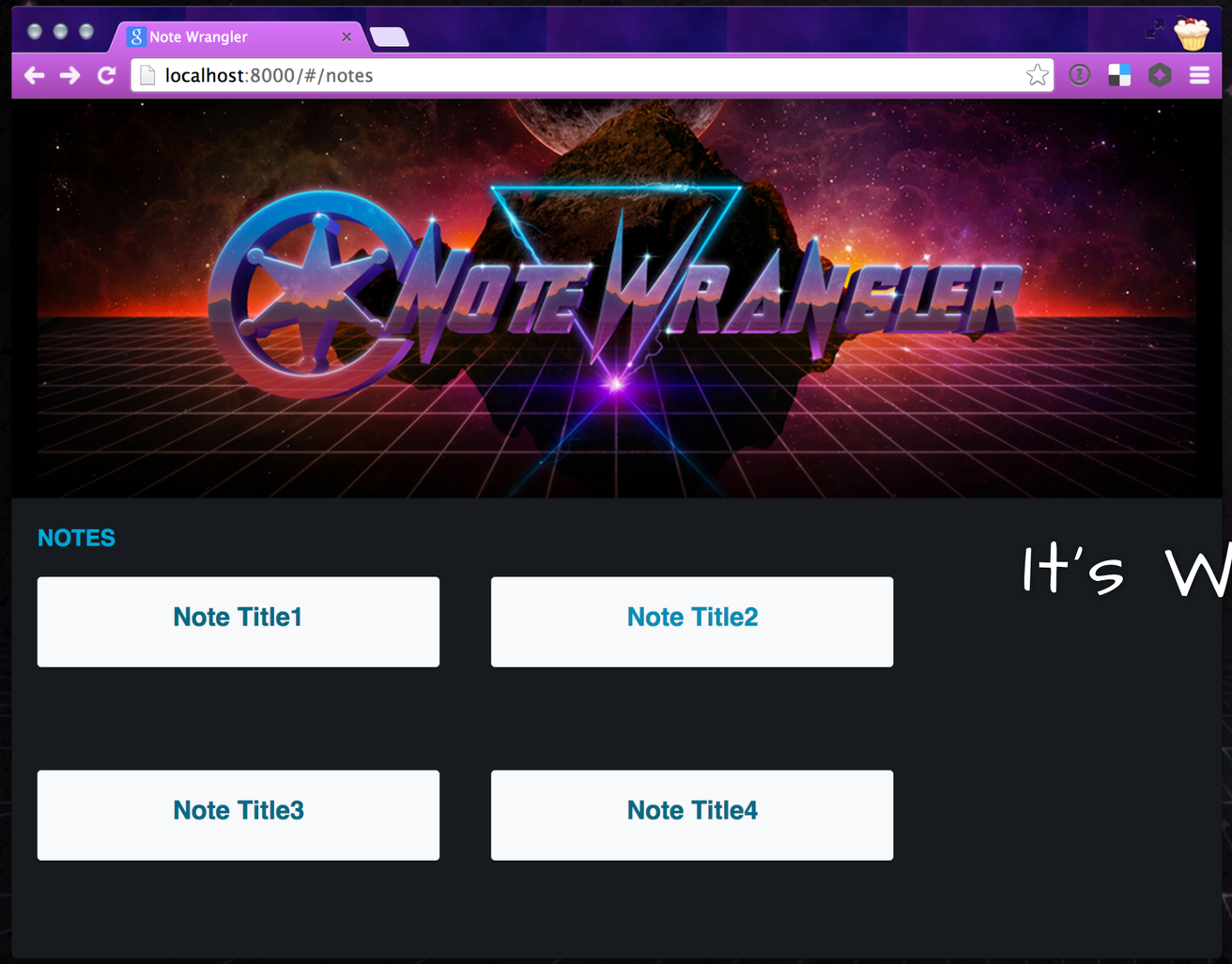
nw-card.js

```
angular.module("NoteWrangler")
.directive("nwCard", function() {
  var num = 1;
  return {
    restrict: "E",
    templateUrl: "templates/directives/nw-card.html",
    scope: {},
    controller: function($scope){
     $scope.header = "Note Title" + num++;
    }
...
```

Give this directive its own scope!

Now With Isolate Scope

With isolate scope our header isn't overriding itself each time the directive is used.



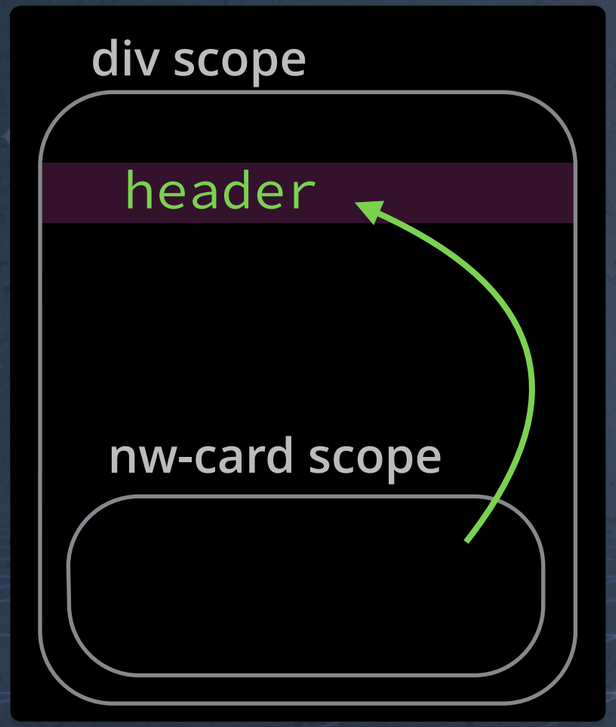
It's WORKING!

Directives and Scope

Directives default

scope: false

Inherited Scope



scope: { }

Isolate Scope

If child adds a new property,
it gets stored on itself.

Child no longer has access to
the parent's scope.

If child adds a new property,
it gets stored on the parent.

STAYING SHARP with Angular.JS

Isolate Scope Has No Access to Parent Scope

We no longer have access to notes from ngRepeat.

notes.html

```
...
<div class="note-wrapper">
  <a class="card-notes" ng-repeat="note in notes" … >
    <nw-card></nw-card>
  </a>
</div>
...
```

nw-card.html

```
<div class="card">
  <h2 class="h3">{{note.header}}</h2>
</div>
```

No access!

How can we use isolate scope and still access our notes array with titles?

Passing Header Into the Directive

We can set the note instance to a header variable and print it out in the template.

notes.html

```
...
<div class="note-wrapper">
  <a class="card-notes" ng-repeat="note in notes" … >
    <nw-card header="{{note.title}}"></nw-card>
  </a>
</div>
...
```

Setting note.title to a header var

nw-card.html

```
<div class="card">
  <h2 class="h3">{{header}}</h2>
</div>
```

Using header var in the template

But to do this, we need an extra step inside our directive.

Back in Our Controller

nw-card.js

```
angular.module("NoteWrangler")
.directive("nwCard", function() {
  var num = 1;
  return {
    restrict: "E",
    templateUrl: "templates/directives/nw-card.html",
    scope: {},
    controller: function($scope){
     $scope.header = "Note Title" + num++;
    }
...
```

We Need a Way to Pass In a Header

```
<nw-card header="{{note.title}}"></nw-card>
```

nw-card.js

```
angular.module("NoteWrangler")
.directive("nwCard", function() {
  var num = 1;
  return {
    restrict: "E",
    templateUrl: "templates/directives/nw-card.html",
    scope: {}
...
```

We need to tell our directive it might receive a header variable and, if so, to bind it to the scope.

Binding a String to Our Scope

We just need a little configuration.

nw-card.js

```
angular.module("NoteWrangler")
.directive("nwCard", function() {
 return {
    restrict: "E",
    templateUrl: "templates/directives/nw-card.html",
    scope: {
      header: "@"
    }
  };
});
```

@ passes in a string

```
<nw-card header="{{note.title}}"></nw-card>
```

There are 3 options when binding data to an isolate scope: @, =, and & characters.

Two-way Binding

So if the value gets changed, it gets changed everywhere.

nw-card.js

```
angular.module("NoteWrangler")
.directive("nwCard", function() {
 return {
    restrict: "E",
    templateUrl: "templates/directives/nw-card.html",
    scope: {
      header: "=" = two-way binds an object
    }
  };
});
```

```
<nw-card header="note.header"></nw-card>
```

No longer need brackets

We Can Add Multiple Bindings

nw-card.js

```
angular.module("NoteWrangler")
.directive("nwCard", function() {
 return {
    restrict: "E",
    templateUrl: "templates/directives/nw-card.html",
    scope: {
      header: "=",
      icon: "="
    }
  };
});
```

Directives Getting Notes Data

We are passing in a header and icon, just like we would pass in variables to a function.

notes.html

```
...
<div class="note-wrapper">
  <a class="card-notes" ng-repeat="note in notes" … >
    <nw-card header="note.header" icon="note.icon"></nw-card>
  </a>
</div>
...
```

nw-card.html

```
<div class="card">
  <h2 class="h3">{{header}}</h2>
  <i class="icon icon-card {{icon}}"></i>
</div>
```

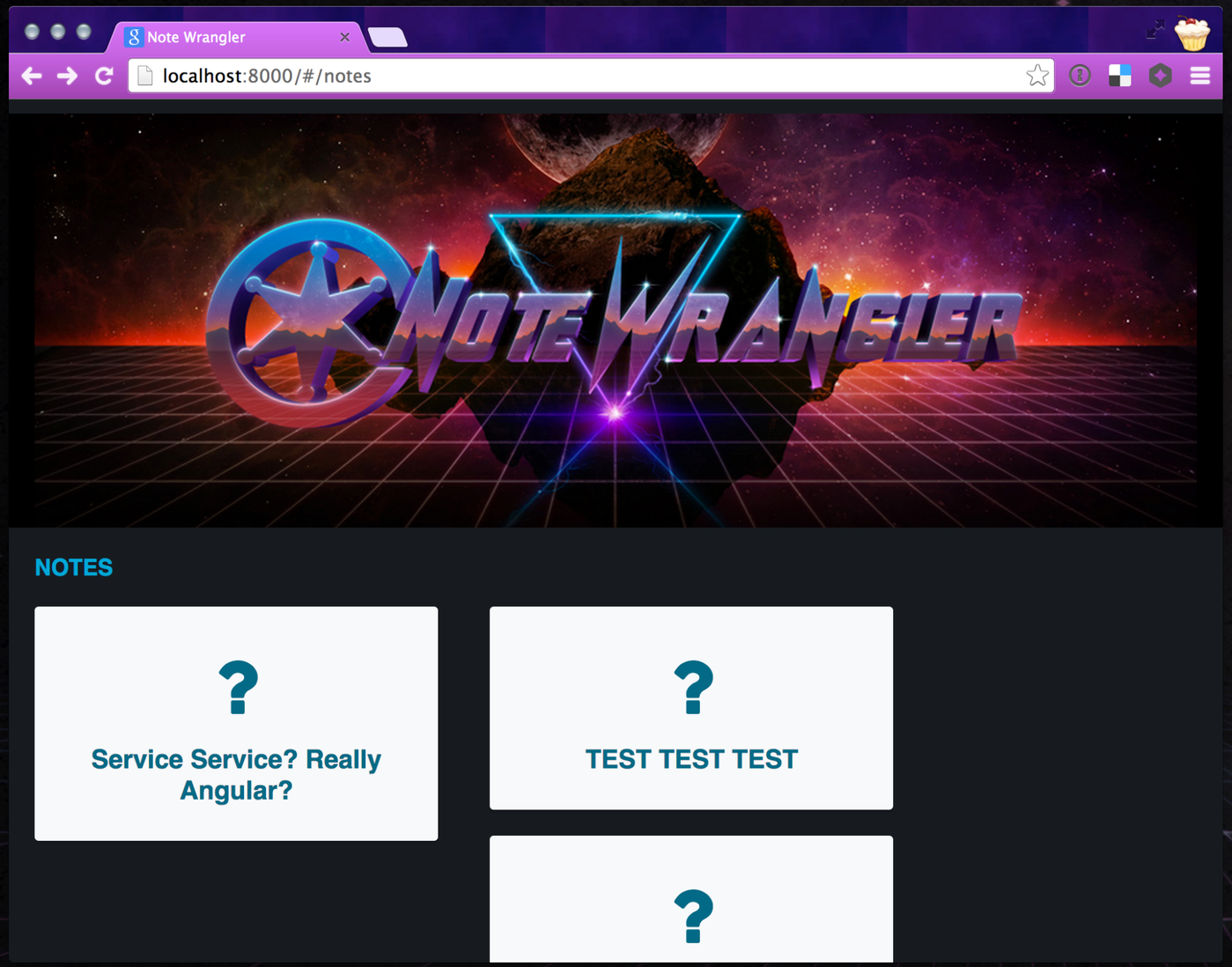
Now Our Directives are Flexy!

```
<nw-card header="note.title" icon="note.icon"></nw-card>
```

Later on we can use this same directive again, but with a different use case.

```
<nw-card header="user.name" icon="user.image"></nw-card>
```

Directives Getting Notes Data



Difference Between $scope and Scope Object

nw-card.js

```
angular.module("NoteWrangler")
.directive("nwCard", function() {
  var num = 1;
  return {
    restrict: "E",
    templateUrl: "templates/directives/nw-card.html",
    scope: {},
    controller: function($scope){
     $scope.header = "Note Title" + num++;
    }
...
```

Allows you to create an isolate scope private to this directive.

Allows you to set values as properties on our scope object.