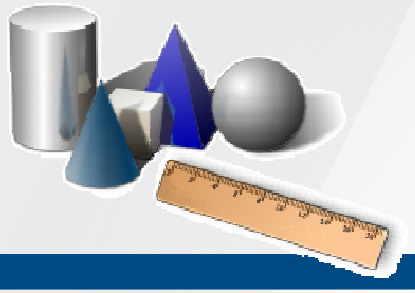
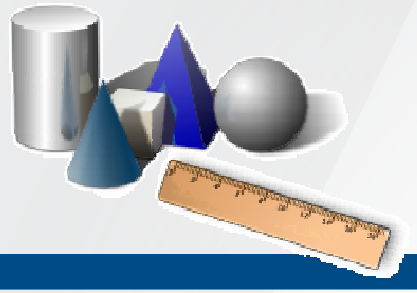


Lab 15

- Get the before project and install it
  - npm install
  - npm start
- The seed comes with all Angular 2 dependencies including angular/router
- Material Design and Angular are very good friends so we could go ahead to make use of Material Design Lite (MDL).
  - Grab MDL with npm:

```
npm install material-design-lite --save
```



Lab 15

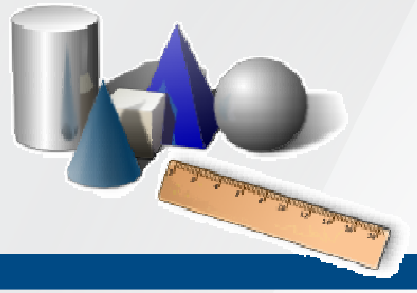
- Add to <link rel="stylesheet" href="styles.css"> in ./index.html MDL files:

```
<!-- MDL CSS library  -->
<link rel="stylesheet" href="/node_modules/material-design-lite/material.min.css">

<!-- MDL JavaScript library  -->
<script src="/node_modules/material-design-lite/material.min.js"></script>

<!-- Material Design Icons  -->
<link rel="stylesheet" href="https://fonts.googleapis.com/icon?family=Material+Icons">
```

- You can then add some custom styles in the styles.css:

➔ You can find the two files in before/src

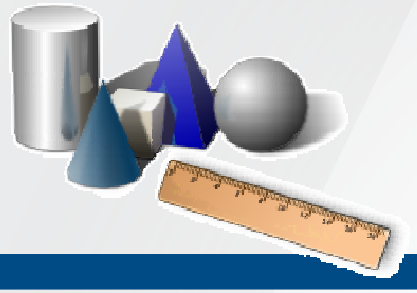
```
/* Jumbotron */
.demo-layout-transparent {
  background: linear-gradient(
      rgba(0, 0, 255, 0.45),
      rgba(0, 0, 255, 0.45)
    ),
    url('assets/scotch-dog1.jpg') center / cover;
  height: 400px;
}

/* Nav Bar */
.demo-layout-transparent .mdl-layout__header,
.demo-layout-transparent .mdl-layout__drawer-button {
  background: rgba(0, 0, 0, 0.3);
  color: white;
}

/* Header Text */
.header-text{
  text-align: center;
  vertical-align: middle;
  line-height: 250px;
  color: white;
}
/* Content Wrapper */
.container{
  width: 80%;
  margin: 0 auto;
}
```



Lab 15

- Defining Routes
  - Let's get started with configuring a basic route
    - We need to create app.routes.ts like this :

```
// Imports
import { ModuleWithProviders }  from '@angular/core';
import { Routes, RouterModule } from '@angular/router';
import { CatListComponent } from './cats/cat-list.component';
import { DogListComponent } from './dogs/dog-list.component';

// Route Configuration
export const routes: Routes = [
  { path: 'cats', component: CatListComponent },
  { path: 'dogs', component: DogListComponent }
];

export const routing: ModuleWithProviders = RouterModule.forRoot(routes);
```

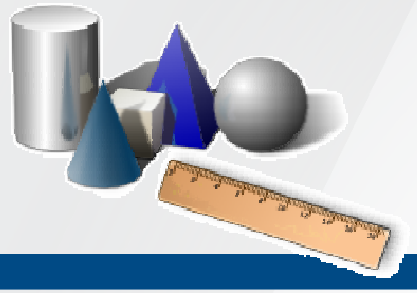


Lab 15

- Creating Placeholder Components :
  - The routes config file imports some components that we need to create.
  - For now we just create them and flesh them out with minimal content, then we can build on them while we move on.
  - First we create ./app/Cats/cat-list.component.ts
  - And ./app/Dogs/dog-list.component.ts

```
import { Component } from '@angular/core';

@Component({
  template: `
    <h2>Cats</h2>
    <p>List of cats</p>`
})

// Component class
export class CatListComponent {}
```

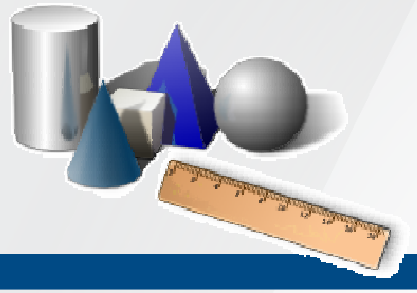
```
import { Component } from '@angular/core';

@Component({
  template: `
    <h2>Dogs</h2>
    <p>List of dogs</p>`
})

// Component class
export class DogListComponent {}
```



Lab 15

- Bootstrapping Our Application
  - Before we bootstrap the app, we need to assemble our imports, providers and declaration using NgModule

```
import { NgModule }      from '@angular/core';
import { BrowserModule } from '@angular/platform-browser';
import { FormsModule }    from '@angular/forms';
import { HttpModule, JsonpModule } from '@angular/http';

// Declarations
import { AppComponent }         from './app.component';
import { CatListComponent }   from './cats/cat-list.component';
//import { CatDetailsComponent }  from './cats/cat-details.component';
import { DogListComponent }      from './dogs/dog-list.component';
//import { DogDetailsComponent }  from './dogs/dog-details.component';
//import { PetService }          from './pet.service';
//import { Pet }          from './pet';
import { routing } from './app.routes';

// Decorator
@NgModule({
  imports: [BrowserModule, FormsModule, HttpModule, JsonpModule, routing],
  declarations: [AppComponent,CatListComponent, DogListComponent],
//  declarations: [AppComponent,CatListComponent, CatDetailsComponent,DogLi
//  providers: [PetService],
  bootstrap: [ AppComponent ]
})

export class AppModule { }
```

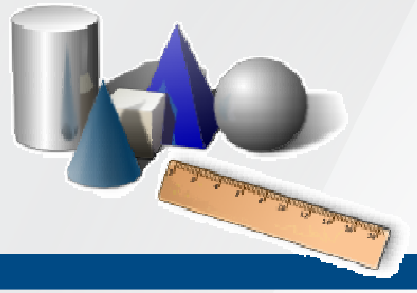


Lab 15

- This is the html template for app.component
  - We can get it in before/src folder
- We must also update the app.component and then start

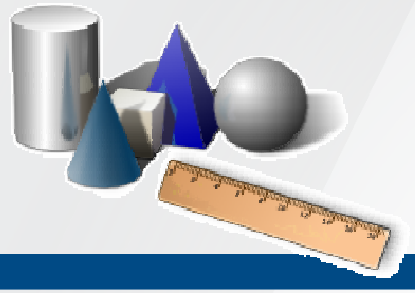
```
import { Component } from '@angular/core';

@Component({
  selector: 'my-app',
  templateUrl : "./app.component.html",

  // Not necessary as we have provided directives
  // `RouterModule` to root module
  // Tell component to use router directives
  // directives: [ROUTER_DIRECTIVES]
})

export class AppComponent  { name = 'Angular'; }
```

```
<div class="demo-layout-transparent mdl-layout mdl-js-layout">
  <header class="mdl-layout__header mdl-layout__header--transparent">
    <div class="mdl-layout__header-row">

      <!-- Title -->
      <span class="mdl-layout-title">Scotch Pets</span>

      <!-- Add spacer, to align navigation to the right -->
      <div class="mdl-layout-spacer"></div>

      <!-- Navigation with router directives-->
      <nav class="mdl-navigation">
        <a class="mdl-navigation__link" [routerLink]="['/']">Home</a>
        <a class="mdl-navigation__link" [routerLink]="['/cats']">Cats</a>
        <a class="mdl-navigation__link" [routerLink]="['/dogs']">Dogs</a>
      </nav>
    </div>
  </header>

  <main class="mdl-layout__content">
    <h1 class="header-text">We care about pets...</h1>
  </main>
</div>

<!-- Router Outlet -->
<router-outlet></router-outlet>
```



Lab 15

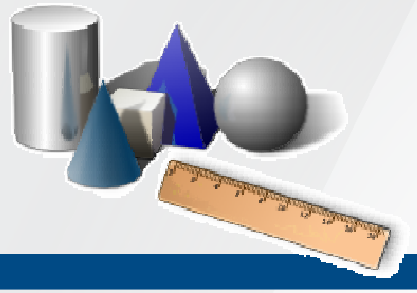
- Going a Little Deeper
  - Yes, we have a functional route, but we all know that real-life applications require a bit more than a simple route. Real apps have index/home page routes for:
    - Landing pages
    - Redirects
    - Route parameters
    - Queries
    - Route restrictions
    - …



Lab 15

- Index/Home Page Route and Redirects
  - First and most important is to fix our index route.
    - We can suppose to redirect to /dogs once the index route is hit.
    - We just successfully killed two birds with one stone.
      - We have an **index route**
      - and we have also seen how we can **redirect to another route**

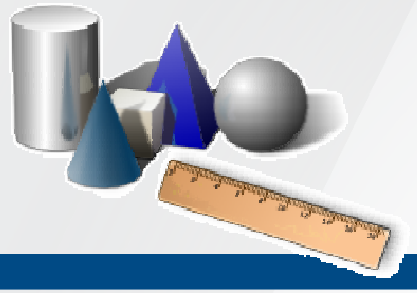
```
// Imports
import { ModuleWithProviders }  from '@angular/core';
import { Routes, RouterModule } from '@angular/router';
import { CatListComponent } from './cats/cat-list.component';
import { DogListComponent } from './dogs/dog-list.component';


// Route Configuration
export const routes: Routes = [
  {
    path: '',
    redirectTo: '/dogs',
    pathMatch: 'full'
  },
  { path: 'cats', component: CatListComponent },
  { path: 'dogs', component: DogListComponent }
];

export const routing: ModuleWithProviders = RouterModule.forRoot(routes);
```



Lab 15

- If you prefer to have a component to the index route, configure as follows:

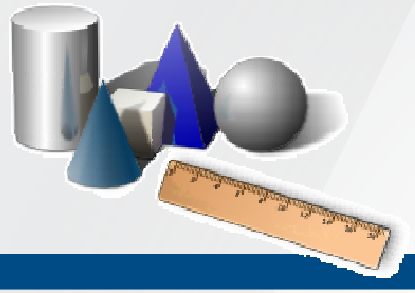
```
export const routes: Routes = [
  {
     path: '',
     component: HomeComponent // Remember to import the Home Component
  },
  { path: 'cats', component: CatListComponent },
  { path: 'dogs', component: DogListComponent }
];
```



Lab 15

- Route Parameters
  - This is a good time to add more features to the demo app by fetching list of pets from a remote server and retrieving each pet details with their ID.
    - This will give us a chance to see how route parameters work.
  - Pet Service to Get Pet Data
    - First we create the Pet class like this pet.ts:

```
export class Pet {
  name: string;
  age: number;
}
```



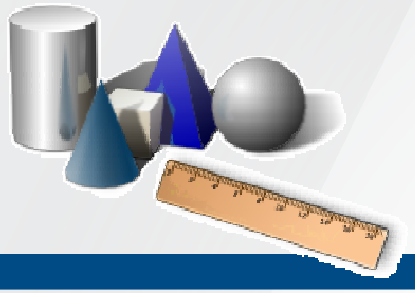
Lab 15

- Add the pet service

```
// Imports
import { Injectable }    from '@angular/core';
import { Jsonp, URLSearchParams } from '@angular/http
import { Pet } from './pet'
import 'rxjs/add/operator/map';

// Decorator to tell Angular that this class can be i
@Injectable()
export class PetService {

  // Class constructor with Jsonp injected
  constructor(private jsonp: Jsonp) { }

  // Base URL for Petfinder API
  private petsUrl = 'http://api.petfinder.com/';


}
```

```
findPets(animal : string) {

  // End point for list of pets:
  // http://api.petfinder.com/pet.find?key=[API_KEY]&animal=[ANIMAL]&format=j
  const endPoint = 'pet.find'
  const API_KEY = '555f8155d42d5c9be4705beaf4cce089'

  // URLSearchParams makes it easier to set query parameters and construct UR
  // rather than manually concatenating
  let params = new URLSearchParams();
  params.set('key', API_KEY);
  params.set('location', 'texas');
  params.set('animal', animal);
```

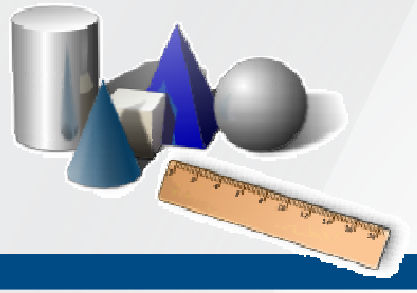
```
// get a pet based on their id
findPetById(id: string) {

  // End point for list of pets:
  // http://api.petfinder.com/pet.find?key=[API_KEY]&animal=[ANIMAL]&for
  const endPoint = 'pet.get'

  // URLSearchParams makes it easier to set query parameters and constru
  // rather than manually concatinating
  let params = new URLSearchParams();
  params.set('key', API_KEY);
  params.set('id', id);
  params.set('format', 'json');
  params.set('callback', 'JSONP_CALLBACK');

  console.log(params);
  // Return response
  return this.jsonp
              .get(this.petsUrl + endPoint, { search: params })
              .map(response => <string[]> response.json().petfinder.pet);
}
```



Lab 15

- Injecting PetService in DogListComponent
  - NB : the associated template is in before/src folder

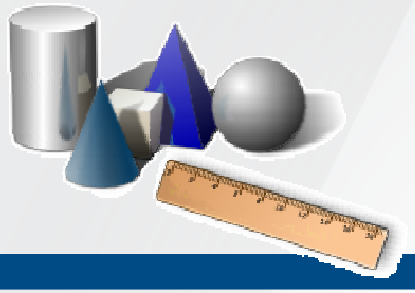
```
import { Component, OnInit } from '@angular/core';
import { PetService } from '../pet.service'
import { Observable } from 'rxjs/Observable';
import { Pet } from '../pet';

@Component({
  templateUrl : "./dog-list.component.html",
})

// Component class implementing OnInit
export class DogListComponent implements OnInit {
  // Private property for binding
  dogs: Observable<string[]>;

  constructor(private petService: PetService) {
  }

  // Load data ones componet is ready
  ngOnInit() {
    // Pass retreived pets to the property
    this.dogs = this.petService.findPets('dog');
  }
}
```



Lab 15

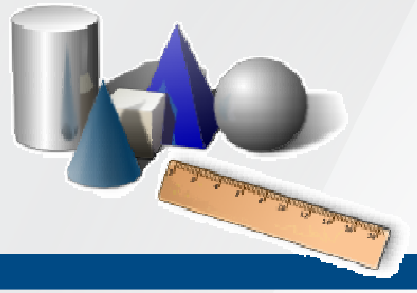
- Let’s make a look to the associated template file
  - In the template file : The trailing .$t is as a result of the API structure and not an Angular thing so you do not have to worry about that
  - We are binding an observable type, dogs to the view and looping through it with the NgFor directive
  - A **VERY** important thing to also note is the routerLink again.
    - This time it does not just point to /dog but has a parameter added

```
<h2>Dogs</h2>
    <p>List of dogs</p>
    <ul class="demo-list-icon mdl-list">
      <li class="mdl-list__item" *ngFor="let dog of dogs | async">
        <span class="mdl-list__item-primary-content">
            <i class="material-icons mdl-list__item-icon">pets</i>
            <a [routerLink]="['/dogs', dog.id.$t]">{{ dog.name.$t }}</a>
        </span>
      </li>
    </ul>
```



Lab 15

- Update the app.module.ts
- Finish the CatListComponent the it looks quite exactly like DogListComponent .

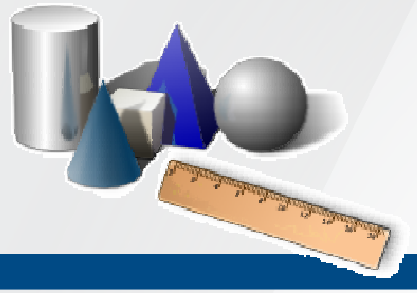


Lab 15

- Details Components
  - The link from the list components points to a non-existing route.
  - The route's component is responsible for retrieving specific pet based on an id.
  - The first thing to do before creating these components is to make there routes
    - Create the dog.routes.ts in app/Dogs and update the app.routes.ts

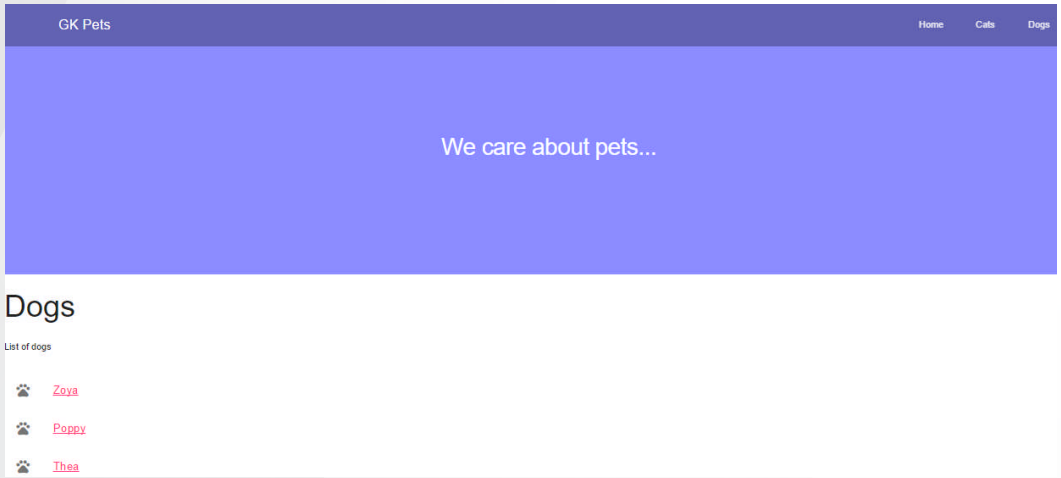
```
import { Routes } from '@angular/router';

import { DogListComponent }    from './dog-list.component';
import { DogDetailsComponent }    from './dog-details.component';

// Route Configuration
export const dogRoutes: Routes = [
  { path: 'dogs', component: DogListComponent },
  { path: 'dogs/:id', component: DogDetailsComponent }
];
```

```
import { ModuleWithProviders }  from '@angular/core';
import { Routes, RouterModule } from '@angular/router';

import { dogRoutes }    from './dogs/dog.routes';

// Route Configuration
export const routes: Routes = [
  {
    path: '',
    redirectTo: '/dogs',
    pathMatch: 'full'
  },
    // Add dog routes form a different file
  ...dogRoutes
];
```



Lab 15

- Add the DogDetailsComponent
- Update the ngmodule
- And enjoy your app :




```
import { Component, OnInit } from '@angular/core';
import { PetService } from '../pet.service'
import { Observable } from 'rxjs/Observable';
import { ActivatedRoute } from '@angular/router';
import { Pet } from '../pet';

@Component({
  template: `
    <div *ngIf="dog">
        <h2>{{dog.name.$t}}</h2>
        <img src="{{dog.media.photos.photo[3].$t}}"/>
        <p><strong>Age: </strong>{{dog.age.$t}}</p>
        <p><strong>Sex: </strong>{{dog.sex.$t}}</p>
        <p><strong>Description: </strong>{{dog.description.$t}}</p>
    </div>
    `
})

// Component class implementing OnInit
export class DogDetailsComponent implements OnInit {
  // Private properties for binding
  private sub:any;
  private dog:string[];

  constructor(private petService: PetService, private route: ActivatedRoute) {
  }

  // Load data ones componet is ready
  ngOnInit() {
      // Subscribe to route params
      this.sub = this.route.params.subscribe(params => {
        let id = params['id'];
       // Retrieve Pet with Id route param
        this.petService.findPetById(id).subscribe(dog => this.dog = dog);
    });
  }

  ngOnDestroy() {
    // Clean sub to avoid memory leak
    this.sub.unsubscribe();
  }
}
```