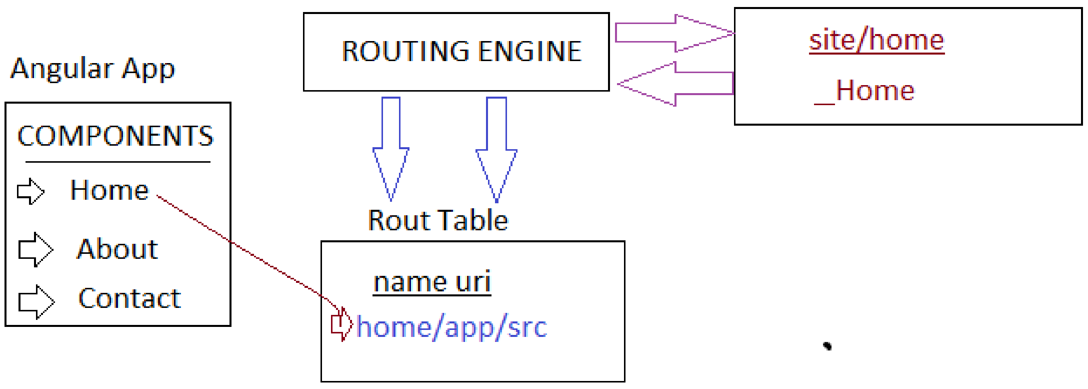
Angular Routing



- Routing is a technique used in web-application to create user and SEO friendly URIs.
- User-friendly URIs allows user to get access to everything from one page.
- It enable SPA architecture where now details are loaded without reloading the complete page.
- Routing uses assembly technique using <u>AJAX</u> calls.
- The SEO friendly URIs allows search engine to identify exact location and type of content in a page, so that it can recommend relative contents.
- Angular routing is a configuration in a rout module, which is loaded in to the application in configuring phase.

  'config phase'
- Routs are initialize in configuring phase.
- Rout table comprises the reference for all variable routs.
- A rout engine is responsible to identify client request and verify the requested URI with rout reference.
- The configuration of angular routs are handled by two classes:-
  a) Routes.
  b) RouterModule.
- They are defined in '@angular/routes' library.

<u>Configuring Routes</u>

- The routs are configured by using a RouterModule that comprises of following details:-

ROUTE PROPERTY	DESCRIPTION
Path	- It specifies the rout name which is used as reference to access any component by using url request.

Component	- It specifies the component name to access and render when any path is requested by client.
redirectTo	- It specifies the default path for redirection automatically on dynamic request.
data	- It provides additional information for routs which includes the outlet name and other details.
pathMatch	- It specifies the url to access when requested path is similar to the existing path.
Outlet	- It specifies the router outlet that is used to render the target details.

- The navigation to any specific component is defined by using ‘routerLink’.
- The target location to render is defined by using <router-outlet>.
- The routes are configured in ‘app-routing.module.ts’.

Synatax:-

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

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


    const routes: Routes = [ ]


    @NgModule({
  imports: [RouterModule.forRoot(routes)],
  exports: [RouterModule]
   })
   export class AppRoutingModule { }
```

Q. Why routes are initialised?

Ans: Routes are defined and loaded in to Rout Table on configuration phase. Hence, initialization is mandatory . Configuration will not allow declaration and rendering.

WildCard Routes:

The ** path in the below route is a wildcard. The router will select this route if the requested URL doesn't match any paths for routes defined earlier in the configuration. This is useful for displaying a "404 - Not Found" page or redirecting to another route.

```
const appRoutes: Routes = [

{ path: '**', component: PageNotFoundComponent }

];
```

- The wildcard route allows to access any content based on dynamic request , it includes the following:-
  {path: ‘ ’, component: ‘if empty’}
  {path: ‘ ** ’, component: ‘if not found’}
  {path: ‘c*t ’, component: ‘if match with wildcard’}

Dynamic Redirection:

- The dynamic redirection can be defined by using the attribute ‘redirectTo’ and ‘pathMatch’.
  Syntax:

  ```
      {path: '', redirectTo: '/home', pathMatch: 'full'}
  ```

  Example:
  1. Add a new angular project.
  >ng new VideoTutorial
  [ignore routing Module]
  2. Add the following components
  >ng g c home
  >ng g c cssdemo
  >ng g c javascript
  >ng g c jquery
  >ng g c contact
  >ng g c notFound
  3. Add new content to those components.
  4. Add routing module in to the project
  >ng g module app-routing --flat
  5. Go to ‘app-routing.module.ts’.

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

const routes: Routes = [
  {path: 'home', component: HomeComponent},
  {path: 'js', component: JavascriptComponent},
  {path: 'jq', component: JqueryComponent},
```

```
  {path: 'contact' , component: ContactComponent},
  {path: '', redirectTo: '/home', pathMatch: 'full'},
  {path: '**', component: NotfoundComponent},
];

@NgModule({
  imports: [RouterModule.forRoot(routes)],
  exports: [RouterModule]
})
export class AppRoutingModule { }
```

6. Go to ‘app.module.ts’

```
imports: [
    AppRoutingModule, BrowserModule
  ],
```

7. Go to ‘app.component.html’

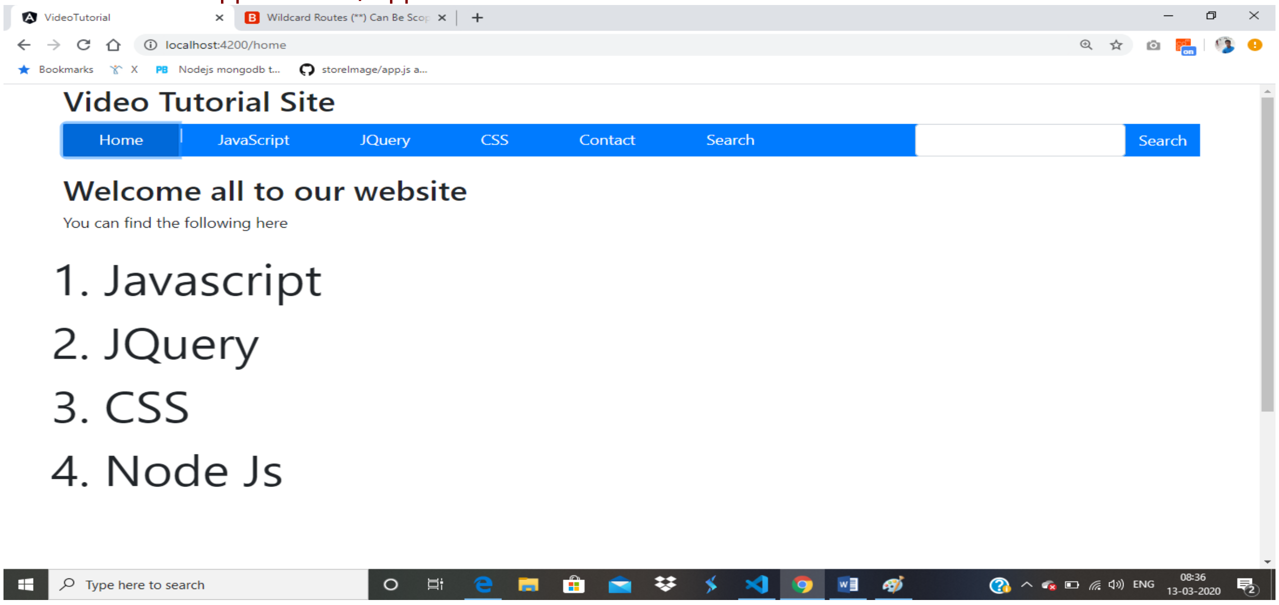
```
<div class="top-bar">
  <div class="container ">
    <h2>Video Tutorial Site</h2>
    <div class="btn-toolbar bg-primary justify-content-between text-white">
      <div class="btn-group col-8 row">
        <a routerLink="/home" class="btn btn-primary">Home</a> <span>|</span>
        <a routerLink="/js" class="btn btn-primary">JavaScript</a>
        <a routerLink="/jq" class="btn btn-primary">JQuery</a>
        <a class="btn btn-primary">CSS</a>
        <a routerLink="/contact" class="btn btn-primary">Contact</a>
        <a routerLink="search/2/tv/23000" class="btn btn-primary">Search</a>
      </div>
      <div class="btn-group">
        <div class="form-inline">
           <input type="text" class="form-control">
           <button class="btn btn-primary">Search</button>
        </div>
      </div>
    </div>
    <div style="height: 700px;margin-top: 20px;">
      <router-outlet></router-outlet>
    </div>
    <div class="bg-primary text-white text-center">
      &copy; Copyright2020
    </div>
  </div>
</div>
```

8. Go to ‘app.module.ts’

```
bootstrap: [Appomponent]
```

9. Go to ‘index.html’

```
<app-root></app-root>
```



Video Tutorial Site

Welcome all to our website

You can find the following here

1. Javascript
2. JQuery
3. CSS
4. Node Js

Route Parameters

- Angular use Route Parameters to transport data from one component to another.
- A route parameter is similar to Querystring.
- It is appended to the browser address bar as a part of url.
- The route parameter are defined in the route configuration by using following techniques.

  { path: ‘routeName/:paramName’ }
- The route parameter are stored in memory with key and value reference.
- You can access the route parameters in any component by using ‘paramMap’ property of ‘activatedRoute’ module.
- Syntax:
- `constructor(private route: ActivatedRoute) { }`

```
this.route.snapshot.paramMap.get('id');
                              .getAll();
                               .keys();
```

- The route parameters can be appended dynamically by using a router link ‘routerLink’.
- Syntax:

  <a routerLink=’path/param1/param2’></a>

  Example:

1. Add a new component to the project
   >ng g c search –spec=false

2. search.component.ts

```
import { Component, OnInit } from '@angular/core';
import {ActivatedRoute} from '@angular/router';

@Component({
  selector: 'app-search',
  templateUrl: './search.component.html',
  styleUrls: ['./search.component.css']
})
export class SearchComponent implements OnInit {
  public productId;
  public productName;
  public productPrice;

  constructor(private route: ActivatedRoute) { }

  ngOnInit() {
    this.productId = this.route.snapshot.paramMap.get('id');
    this.productName = this.route.snapshot.paramMap.get('name');
    this.productPrice = this.route.snapshot.paramMap.get('price');
  }

}
```

3. **search.component.html**

```
<div class="container">
    <dl>
        <dt>Product Id</dt>
        <dd>{{productId}}</dd>
        <dt>Product Name</dt>
        <dd>{{productName}}</dd>
        <dt>Product Price</dt>
        <dd>{{productPrice}}</dd>
    </dl>
</div>
```

4.**Go to 'app-routing.module.ts'**

```
{path: 'search/:id/:name/:price', component: SearchComponent},
```

5. Go to 'app.component.html'

```
<a routerLink="search/2/tv/23000" class="btn btn-primary">Search</a>
```

Add a service to the current project

>ng g s data –spec=false

data.service.ts

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

@Injectable()
export class DataService {

  constructor() { }
  public GetCategories() {
     return [
       {CategoryId: 1, CategoryName: 'Electronics'},
       {CategoryId: 2, CategoryName: 'Shoes'},
     ];
  }
  public GetProducts() {
    return [
      {ProductId: 1, Name: 'Samsung TV', Price: 23000.55, CategoryId: 1, Mfd:
new Date('2019/01/20')},
      {ProductId: 2, Name: 'Nike Casuals', Price: 3000.55, CategoryId: 2, Mfd:
 new Date('2019/05/10')},
      {ProductId: 3, Name: 'MI Mobile', Price: 10000.55, CategoryId: 1, Mfd: n
ew Date('2020/05/20')},
      {ProductId: 4, Name: 'Lee cooper Boot', Price: 5000.55, CategoryId: 2, M
fd: new Date('2019/12/24')},
    ];
  }
}
```

Register the service in the app.module.ts

```
               providers: [DataService]
Add following components to the project
 >ng g c categorieslist -spec=false
 >ng g c productslist -spec=false
```

Categorieslist.component.ts

```
import { Component, OnInit } from '@angular/core';
import { DataService } from '../data.service';

@Component({
  selector: 'app-categorylist',
  templateUrl: './categorylist.component.html',
  styleUrls: ['./categorylist.component.css']
})
export class CategorylistComponent implements OnInit {
  public categories = [];

  constructor(private data: DataService) { }

  ngOnInit() {
    this.categories = this.data.GetCategories();
  }
public CategoryClick(item) {
  this.router.navigate(['categories', item.CategoryId]);
}
}
```

Categorieslist.component.html

```
<div class="container">
    <h2>Categories Lists</h2>
    <ol>
        <li *ngFor="let item of categories">
            <a routerLink="{{item.CategoryId}}">{{item.CategoryName}}</a>
        </li>
    </ol>
</div>
```

Productslist.component.ts

```
import { ActivatedRoute } from '@angular/router';
import { DataService } from './../data.service';
import { Component, OnInit } from '@angular/core';

@Component({
  selector: 'app-productslist',
  templateUrl: './productslist.component.html',
  styleUrls: ['./productslist.component.css']
})
public categoryId;
public products = [];
  constructor(private data: DataService, private route: ActivatedRoute) { }

  ngOnInit() {
    this.categoryId = this.route.snapshot.paramMap.get('id');
    this.products = this.data.GetProducts().filter(x => x.CategoryId == this.c
ategoryId);
  }
}
```

Productslist.component.html

```
<h2>Products List</h2>
<ol>
    <li *ngFor="let item of products">
    {{item.Name}} - {{item.Price}}
    </li>
</ol>
```

App-routing.module.ts

```
{path: 'categories', component: CategorylistComponent},
 {path: 'categories/:id', component: ProductslistComponent}
```

Dynamically Navigating on Button Click

Go to 'categorieslist.component.ts'

```
import { RouterModule, Router } from '@angular/router';
import { Component, OnInit } from '@angular/core';
import { DataService } from '../data.service';

@Component({
  selector: 'app-categorylist',
  templateUrl: './categorylist.component.html',
  styleUrls: ['./categorylist.component.css']
})
export class CategorylistComponent implements OnInit {
  public categories = [];

  constructor(private data: DataService, private router: Router) { }

  ngOnInit() {
    this.categories = this.data.GetCategories();
  }
public CategoryClick(item) {
  this.router.navigate(['categories', item.CategoryId]);
}
}
```

Go to 'categorieslist.component.html'

```
<div class="container">
    <h2>Categories Lists</h2>
    <ol>
        <li *ngFor="let item of categories">
<button (click)="CategoryClick(item)" class="btn btnlink">
{{item.CategoryName}}
</button>
        </li>
    </ol>
</div>
```

```
Router.navigation([‘path’, params])
```

Child Routes

- A child route represents routes within the given context.
- It is configuring of route within existing route.
- It is defined by using children attribute.
- Synatax:

```
{
Path: ‘parent’, component: parentComponent,
Children: [
        {
          Path: ‘child’ , component: childComponent
```

```
            }
          ]
        }
```

- The redirection for child route within the parent is defined by using the “relativeTo” attribute in router.navigate() .
- Syntax:
  This.router.navigate([‘childPath’, params],
                {relativeTo:this.route})

  Example:
  1. Add a new component
     >ng g c productdetails –spec= false

  2.Go to ‘app-routing.module.ts’

```
    {path: 'categories/:id', component: ProductslistComponent,

  children: [
    {path: 'details/:id', component: ProductdetailsComponent}
  ]
}
```

3. Go to ‘productlist.component.ts’

```
  public GetDetails(item) {

    this.router.navigate(['details', item.ProductId], {relativeTo: this.route}
);
  }
```

4. Go to ‘productlist.component.html’

```
<h2>Products List</h2>
<ol>
    <li *ngFor="let item of products">
    <!-- {{item.Name}} - {{item.Price}} -->
    <button (click)="GetDetails(item)" class="btn btn-
link">{{item.Name}}</button>
    </li>
</ol>
<div>
    <h2>Product Details</h2>
    <router-outlet></router-outlet>
</div>
```

5. Go to ‘productdetails.component.ts’

```
import { DataService } from './../data.service';
import { ActivatedRoute } from '@angular/router';
import { Component, OnInit } from '@angular/core';


@Component({
  selector: 'app-productdetails',
  templateUrl: './productdetails.component.html',
  styleUrls: ['./productdetails.component.css']
})
export class ProductdetailsComponent implements OnInit {
  public  productid;
  public products = [];

  constructor(private route: ActivatedRoute, private data: DataService) { }

  ngOnInit() {
    this.productid= this.route.snapshot.paramMap.get('id');
    this.products= this.data.GetProducts().filter(x => x.ProductId == this.pro
ductid);
  }

}
```

6. Go to 'productdetails.component.html'

```
<table class="table table-hover">
    <thead>
        <th>Name</th>
        <th>Price</th>
        <th>Manufactured</th>
    </thead>
    <tbody *ngFor="let prod of products">
        <td>{{prod.Name}}</td>
        <td>{{prod.Price | currency:'INR'}}</td>
        <td>{{prod.Mfd | date}}</td>
    </tbody>
</table>
```