Angular: organizing files and folders of an app

Angular is a web development framework that offers a robust framework for building modern, scalable applications. Proper organization of files and directories is essential to keeping an Angular project manageable, understandable, and maintainable over time. In this guide, we'll explore best practices for structuring and organizing files and directories in an Angular app.

Basic Structure of the Project

Let's start with the basic structure of the Angular project. Creating an orderly structure is critical to the clarity and maintainability of your code. The basic structure might look like this:

```
/app
   /core
     - app.component.ts
     - app.module.ts
     - ...
   /shared
     /components
     /models
     /services
     - shared.module.ts
   /features
     /feature1
       /components
       /models
```

```
        /services
        - feature1.module.ts
      /feature2
        ...
    - main.ts
/assets
/environments
```

- The `core` folder contains the app's core module (`app.module.ts`), the core component (`app. component.ts`) and other essential files.
- The `shared` folder contains modules, components, services and templates shared between different parts of the 'application.
- The `features` folder hosts the various feature modules of the application. Each feature should have its internal structure similar to that of the main folder.
- The `assets` folder is dedicated to static assets such as images or JSON files.
- The `environments` folder contains configuration files for development, production, etc. environments.

Angular modules

Modules in Angular help organize code into functional units. Each app feature should have its own module. Within each module, you can further organize files into folders such as `components`, `services`, and `models`.

```
// Example of a module structure

/feature1
```

```
   /components
     - feature1.component.ts
   /models
     - feature1.model.ts
   /services
     - feature1.service.ts
   - feature1.module.ts
```

Shared services and components

The `shared` folder is crucial for components, services, and templates that are used in multiple parts of the application. Create a shared module (`shared.module.ts`) to make it easier to import these elements into other parts of the app.

Routing

For complex Angular applications, it is recommended to organize routing in a dedicated folder. For example:

```
/app
   /core
     /routing
       - app-routing.module.ts
```

Test

Although testing is a separate aspect, organizing tests in a structure similar to that of the main source code can simplify long-term maintenance.

```
/src
   /app
     ...
   /testing
     /unit
       /components
       /services
     /e2e
```

Conclusions

Proper organization of files and directories in an Angular app is critical to ensuring scalability, maintainability, and understandability of the code over time. By following best structuring practices, you can create an orderly and easily manageable development environment. Remember that these guidelines can be adapted based on the specific needs of the project, but it is always a good practice to maintain consistency within the development team.