academeez

# Workspace

## Using @angular/cli for building a workspace

# Workspace

- A workspace can hold multiple projects and libraries
- According to the docs: “Beginners and intermediate users are encouraged to use **ng new** to create a separate workspace for each application.”
- According to the docs: “Angular also supports workspaces with multiple projects. This type of development environment is suitable for advanced users who are developing shareable libraries, and for enterprises that use a "monorepo" development style, with a single repository and global configuration for all Angular projects.”

# Workspace EX

- If you are looking to start your next project and arrange it in a more workspace suitable style we recommend using the following command:
  - npx @angular/cli new workspace-name --create-application false
- This command will not create a default project.

# Workspace EX - create project

academeez

- To create a new project in your workspace use the following command:
  - npx ng g application name-of-app
- You can create multiple app in the same workspace, there are organisations that maintain a single monorepo in this style for all their angular apps
- You can run a development server for a specific app
  - npx ng serve name-of-app
- You can build a specific app with the command:
  - npx ng build name-of-app —prod
- The build command will create a folder in the **dist** folder for every app

# Workspace EX - create library

academeez

- Working with workspace means splitting your projects to libraries
- A library will hold reusable modules, components, services, directives, pipes, that are pluggable to multiple projects
- To create a library:
  - npx ng g library name-of-lib
- You will see a new folder is created in the **projects** folder that holds the code for the library you created.
- To build the library run:
  - npx ng build name-of-lib --prod

# paths

- After creating the library you will notice that @angular/cli modified the file: **tsconfig.base.json**
- This is what was added to the **tsconfig.base.json**

```
"paths": {
  "name-of-lib": [
    "dist/name-of-lib/name-of-lib",
    "dist/name-of-lib"
  ]
}
```

# paths - ex

academeez

- This configuration can be translated to this:
  - If I find an import to **name-of-lib** I will resolve that import with this array
- During development of the library might be easier to point to the **projects/name-of-lib/src/public-api.ts**

```
"paths": {
  "name-of-lib": [
    "dist/name-of-lib/name-of-lib",
    "dist/name-of-lib"
  ]
}
```

# library ex

- After changing the **paths** in the **tsconfig.base.json** create a component in the library and use that component in the project you created.
- Create another project and use the same component in another project
- The component in the library should just display text

# Manage dependencies - ex

academeez

- You now have 2 projects using a component from a shared library.
- It is recommended to use the trick with **paths** pointing to the library src code only while developing the library
- Once the library is shared by multiple projects, it is recommended to use NPM to manage the versions of the library
- Build the library you created
- Try to push the library to NPM

# Manage dependencies - ex

academeez

- After pushing your package, the projects will start using the package from npm and not from the src code of the library.
- You can modify the **paths** in **tsconfig.base.json** to the following:
- Now you can leverage both npm and also use the src code if you want the nightly

```
"paths": {
  "name-of-lib@dev": [
    "./projects/name-of-lib/src/public-api.ts"
  ]
}
```

# Monorepo

academeez

- It is common for organisation to organise in a similar manner their entire angular workspace in once place
- All the developers enhance a single workspace which makes it easier to view all the code and share libraries among teams
- It is easier to develop and share organisation tools for forms, authentication, styles, etc.
- You need to manage one node_modules
- Manage one testing environment
- Updating angular is done with one command for all the projects
- There is additional challenge with managing libraries dependencies and build

# Summary

- @angular/cli is not only used for creating an angular project, it can also be used for managing a workspace containing multiple projects and libraries
- It’s encouraged to build libraries to share logic across multiple projects
- You can use NPM to manage the version of the package

academeez

# Thank You

## Next Lesson: Ivy renderer