

# Angular Modules Handouts

## Angular Modules

A module encapsulates a group of components, services, pipes, directives, services, etc that are responsible to satisfy a specific cohesive set of business capabilities.

an Angular app is bootstrapped by a root module. Other modules can be imported to the root or to each other as children by the hierarchy of any depth.

- To create a module run: `ng g module <module name>`
- To import a module to another one as a child, add the child module to the imports of the parent module.

```
@NgModule({  
  ...  
  
  ,  
  imports: [  
    TaskListModule  
  ],  
})  
  
export class AppModule { }
```

```
@NgModule({  
  declarations: [  
    TaskListComponent  
  ],  
  exports: [  
    TaskListComponent  
  ],  
  ...  
})  
  
export class TaskListModule { }
```

The diagram illustrates the relationship between three Angular modules. On the left, the `AppModule` is defined with `TaskListModule` in its `imports` array. On the right, the `TaskListModule` is defined with `TaskListComponent` in its `declarations` array and `TaskListComponent` in its `exports` array. A blue arrow points from the `TaskListModule` class declaration in the right-hand code block to the `TaskListModule` entry in the `imports` array of the `AppModule` in the left-hand code block. Another yellow arrow points from the `TaskListComponent` entry in the `declarations` array to the `TaskListComponent` entry in the `exports` array of the right-hand code block.

- To declare a component in a module add the component's class to the `declarations` of the `@Module`.

- To expose components of a module to parent modules you should add the component to the `exports` array of the `@Module`.
- To provide a service to a module add the service to the `providers` array of the `@Module`:

```
@NgModule({  
  ...  
  providers: [ TaskService, ... ]  
})  
export class TaskModule { }
```

