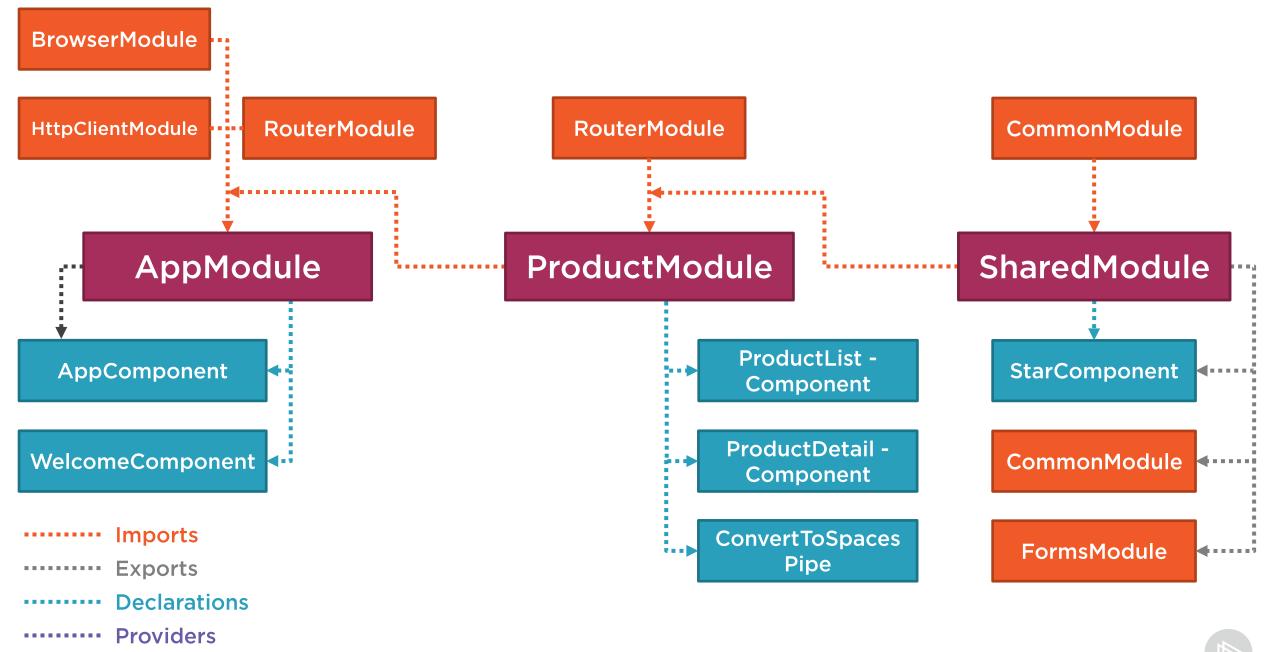


Deborah Kurata CONSULTANT | SPEAKER | AUTHOR | MVP | GDE @deborahkurata | blogs.msmvps.com/deborahk/





Bootstrap



### Module Overview



What Is an Angular Module?
Angular Module Metadata
Creating a Feature Module
Defining a Shared Module
Revisiting AppModule



#### What Is an Angular Module?

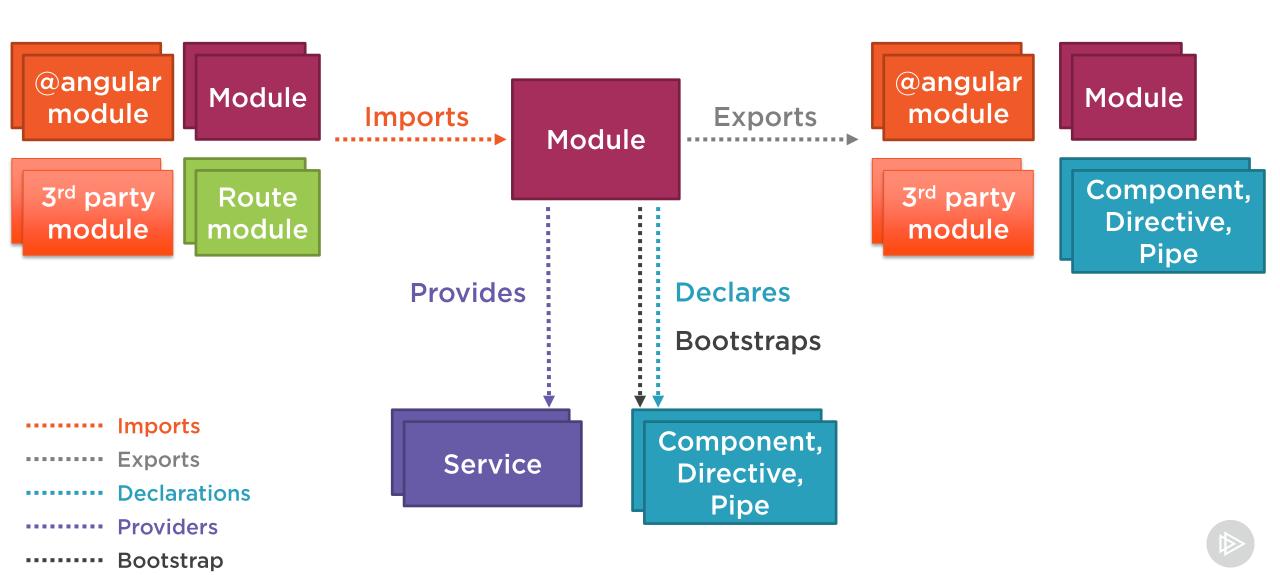


#### A class with an NgModule decorator

#### Its purpose:

- Organize the pieces of our application
- Arrange them into blocks
- Extend our application with capabilities from external libraries
- Provide a template resolution environment
- Aggregate and re-export





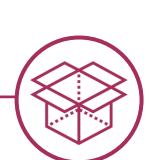
ProductList - Component

ProductDetail - Component

WelcomeComponent

```
...
<a [routerLink]="['/welcome']">
        Home</a>
<a [routerLink]="['/products']">
        Product List</a>

...
<router-outlet>
...
```



**Angular Module** 

RouterModule

RouterModule

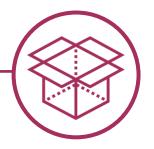
ProductList - Component

ProductDetail - Component

WelcomeComponent

```
...
<input type='text'
        [(ngModel)]='listFilter' />
...
```

Module



RouterModule

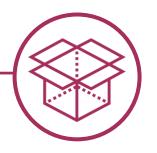
ProductList - Component

ProductDetail - Component

WelcomeComponent

FormsModule

3rowserModule



ConvertToSpaces Pipe

RouterModule

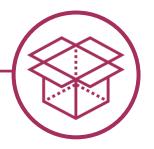
ProductList - Component

ProductDetail - Component

WelcomeComponent

FormsModule

BrowserModule





ConvertToSpaces Pipe

RouterModule

ProductList - Component

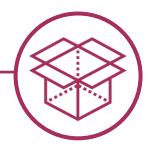
ProductDetail - Component

WelcomeComponent

```
codule

cpm-star
  [rating]='product.starRating'
  (ratingClicked)='onRatingClicked($event)'>

codule
```



ProductList - Component

ProductDetail - Component

WelcomeComponent

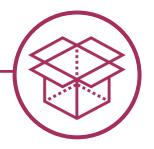
ConvertToSpaces Pipe

StarComponent

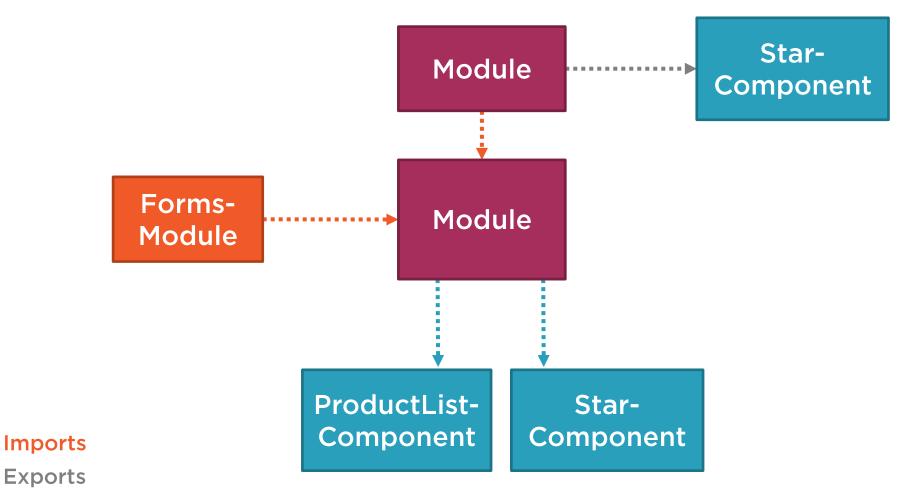
RouterModule

FormsModule

BrowserModule



#### Template Resolution Environment



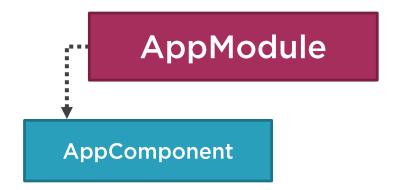
**Declarations** 

**Providers** 

Bootstrap



#### Bootstrap Array



```
app.module.ts
```

```
bootstrap: [ AppComponent ]
```



### Bootstrap Array Truth #1

Every application must bootstrap at least one component, the root application component.

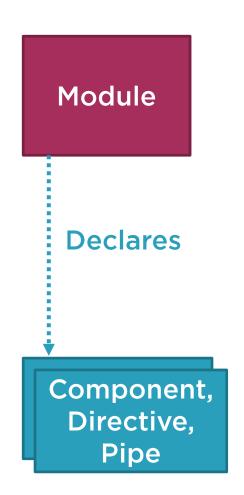


## Bootstrap Array Truth #2

The bootstrap array should only be used in the root application module, AppModule.



#### Declarations Array



#### app.module.ts

```
declarations: [
    AppComponent,
    WelcomeComponent,
    ProductListComponent,
    ProductDetailComponent,
    ConvertToSpacesPipe,
    StarComponent
]
```

**Declarations** 



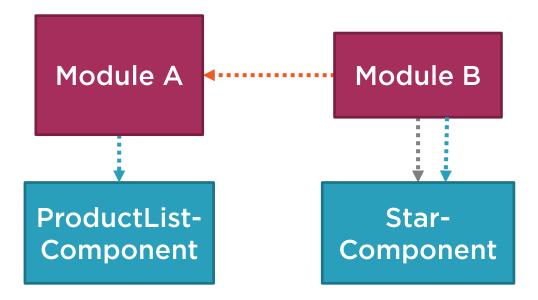
Every component, directive, and pipe we create must belong to one and only one Angular module.



Only declare components, directives and pipes.



Never re-declare components, directives, or pipes that belong to another module



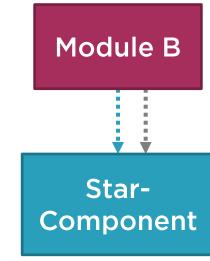




All declared components, directives, and pipes are

private by default.

They are only accessible to other components, directives, and pipes declared in the same module.







The Angular module provides the template resolution environment for its component templates.

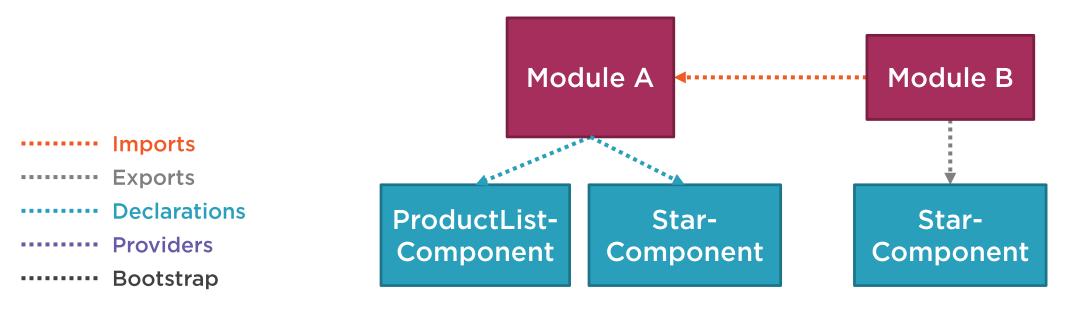
#### product-list.component.html

```
<pm-star ...>
</pm-star>
```

#### star.component.ts

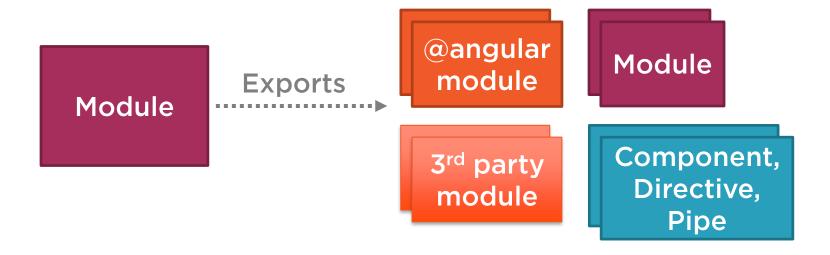
```
@Component({
    selector: 'pm-star',
    template: ...
})
```

The Angular module provides the template resolution environment for its component templates.





#### Exports Array



Imports
Exports
Declarations
Providers
Bootstrap



Export any component, directive, or pipe if other components need it.

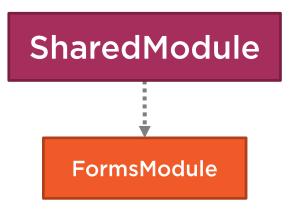


Re-export modules to re-export their components, directives, and pipes.



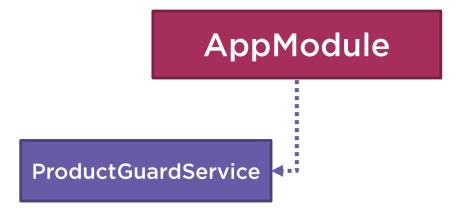
We can re-export something without importing it first.







Never export a service.



ImportsExportsDeclarationsProvidersBootstrap



### @angular Module module 3<sup>rd</sup> party Route module module **Imports** Module

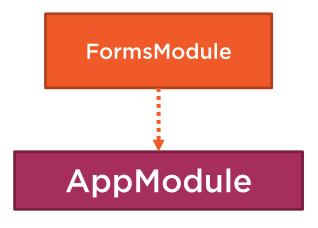
#### Imports Array

#### app.module.ts

```
imports: [
  BrowserModule,
  FormsModule,
  HttpClientModule,
  RouterModule.forRoot([...])
]
```



Importing a module makes available any exported components, directives, and pipes from that module.



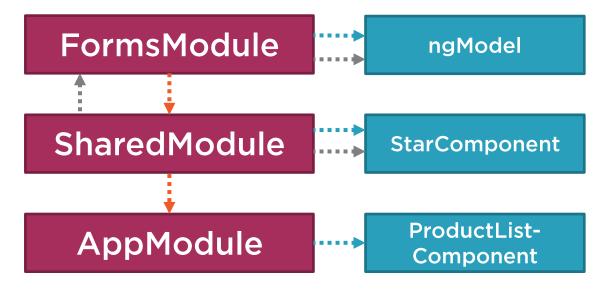




Only import what this module needs.

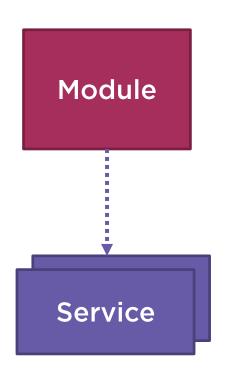


Importing a module does NOT provide access to its imported modules





#### Providers Array



```
ImportsExportsDeclarationsProvidersBootstrap
```

#### app.module.ts

```
providers: [ ProductService ]
...
```

#### product.service.ts

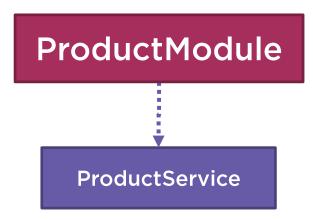
```
...
@Injectable({
  providedIn: 'root'
})
export class ProductService {
...
```



## Providers Array Truth #1

Any service provider added to the providers array is registered at the root of the application.





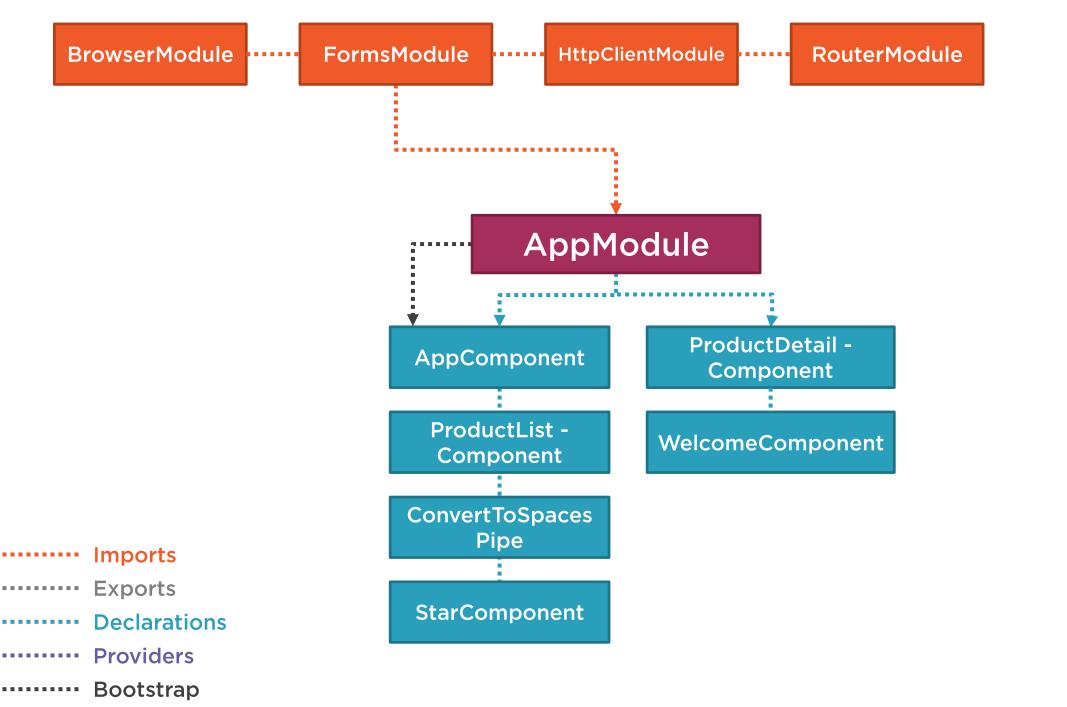


## Providers Array Truth #2

Don't add services to the providers array of a shared module.

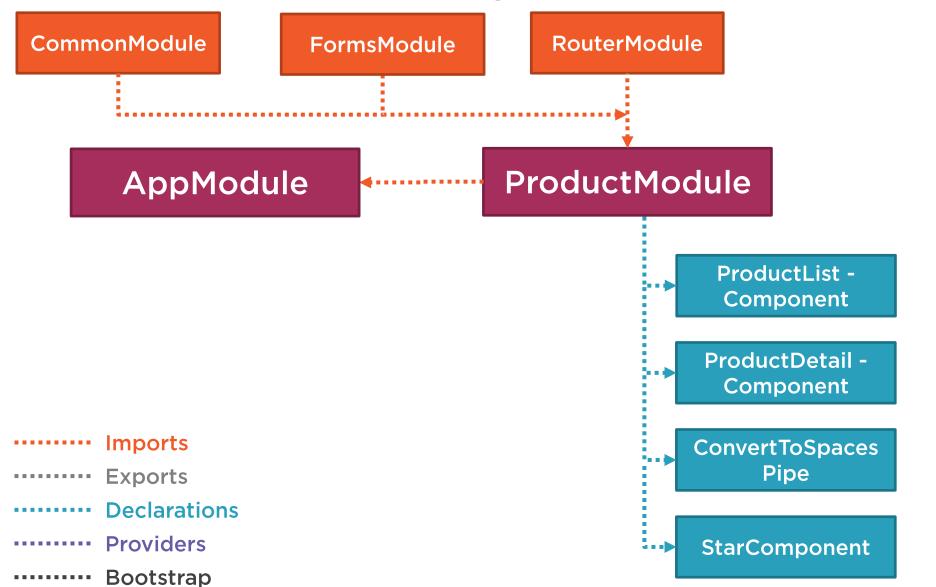
Consider building a CoreModule for services and importing it once in the AppModule.







### Defining a Feature Module





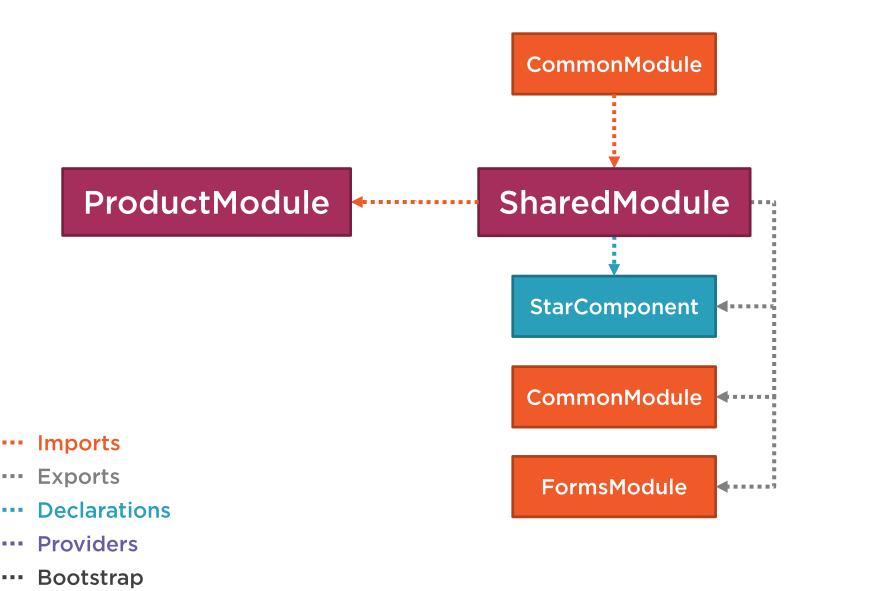
# Demo



**Building a Feature Module** 

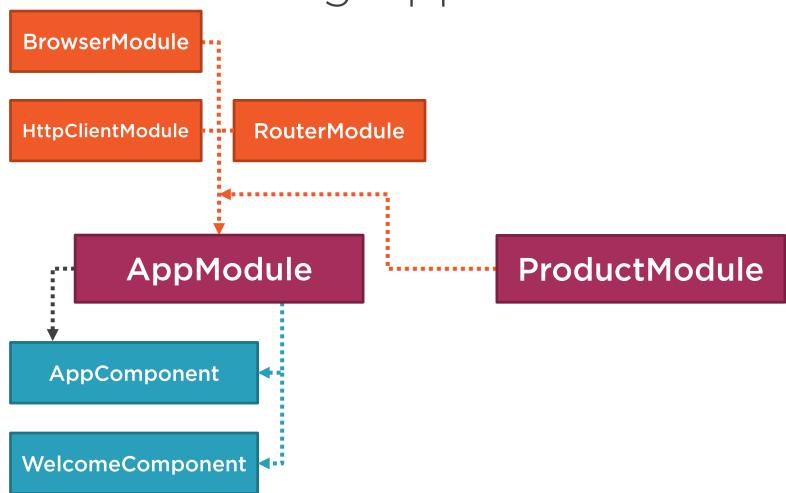


# Defining a Shared Module





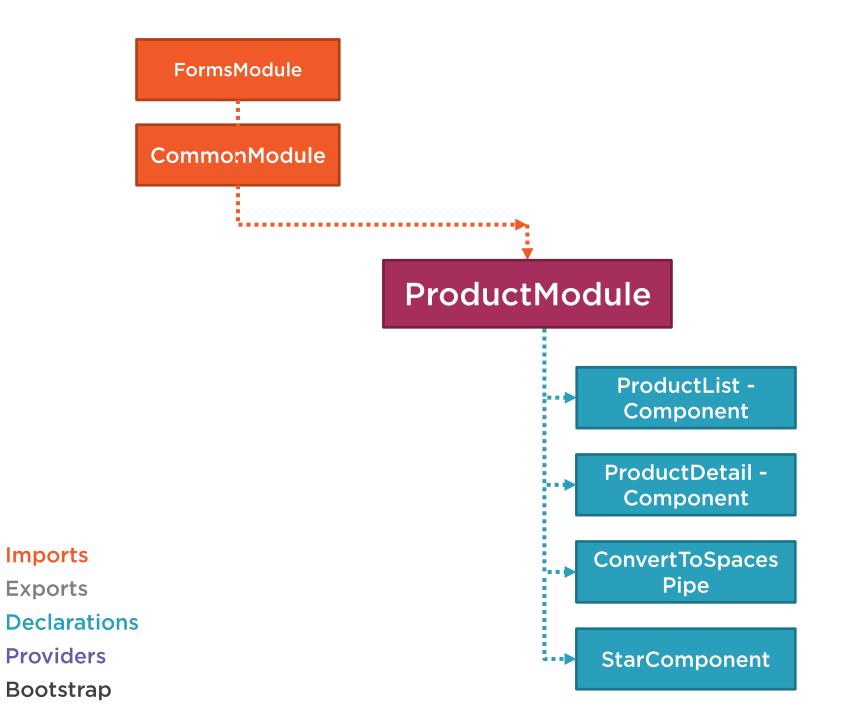
# Revisiting AppModule



Exports
Declarations
Providers
Bootstrap

**Imports** 







### Application Routing Module

#### app-routing.module.ts

```
import { NgModule } from '@angular/core';
import { RouterModule } from '@angular/router';
import { WelcomeComponent } from './home/welcome.component';
@NgModule({
  imports: [
    RouterModule.forRoot([
      { path: 'welcome', component: WelcomeComponent },
      { path: '', redirectTo: 'welcome', pathMatch: 'full'},
      { path: '**', redirectTo: 'welcome', pathMatch: 'full' }
 exports: [ RouterModule ]
export class AppRoutingModule { };
```



### Using the Routing Module

#### app.module.ts

```
@NgModule({
  imports: [
    BrowserModule,
    HttpClientModule,
    ProductModule,
    AppRoutingModule
  declarations: [ AppComponent, WelcomeComponent ],
  bootstrap: [ AppComponent ]
})
export class AppModule { }
```

### Using the Routing Module

#### app.module.ts

```
@NgModule({
  imports: [
    BrowserModule,
    HttpClientModule,
    AppRoutingModule,
    ProductModule
  declarations: [ AppComponent, WelcomeComponent ],
  bootstrap: [ AppComponent ]
})
export class AppModule { }
```

### Feature Routing Module

#### product-routing.module.ts

```
import { NgModule } from '@angular/core';
import { RouterModule } from '@angular/router';
import { ProductListComponent } from './product-list.component';
import { ProductDetailComponent } from './product-detail.component';
import { ProductDetailGuard } from './product-detail.guard';
@NgModule({
 imports: [
    RouterModule.forChild([
      { path: 'products', component: ProductListComponent },
      { path: 'products/:id', canActivate: [ ProductDetailGuard],
        component: ProductDetailComponent }
  exports: [ RouterModule ]
export class ProductRoutingModule { };
```

#### Using the Routing Module

#### product.module.ts

```
@NgModule({
  imports: [
    SharedModule,
    ProductRoutingModule
  declarations: [
    ProductListComponent,
    ProductDetailComponent,
    ConvertToSpacesPipe
export class ProductModule {}
```



#### Angular Module Checklist: Module Structure



Root application module (AppModule)

Feature modules

Shared module (SharedModule)

**Routing modules** 



## Angular Module Checklist: NgModule Metadata



**Bootstrap: Startup component(s)** 

Declarations: What belongs to this module

Exports: What an importing module can use

Imports: Supporting modules

**Providers: Service providers** 

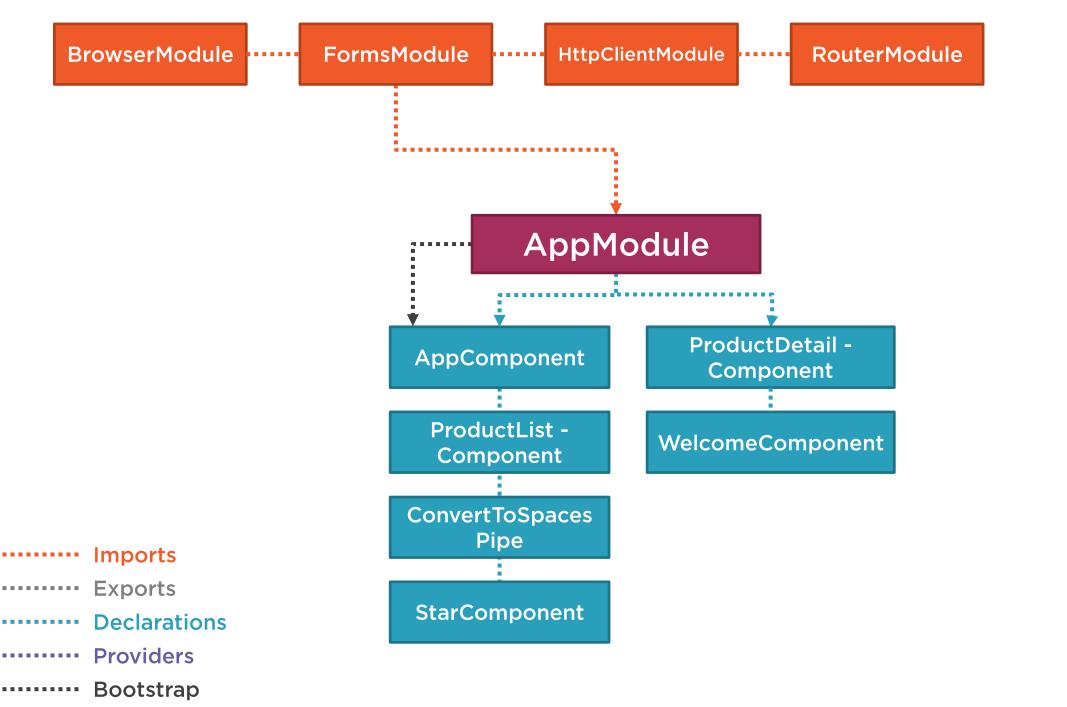


### Summary

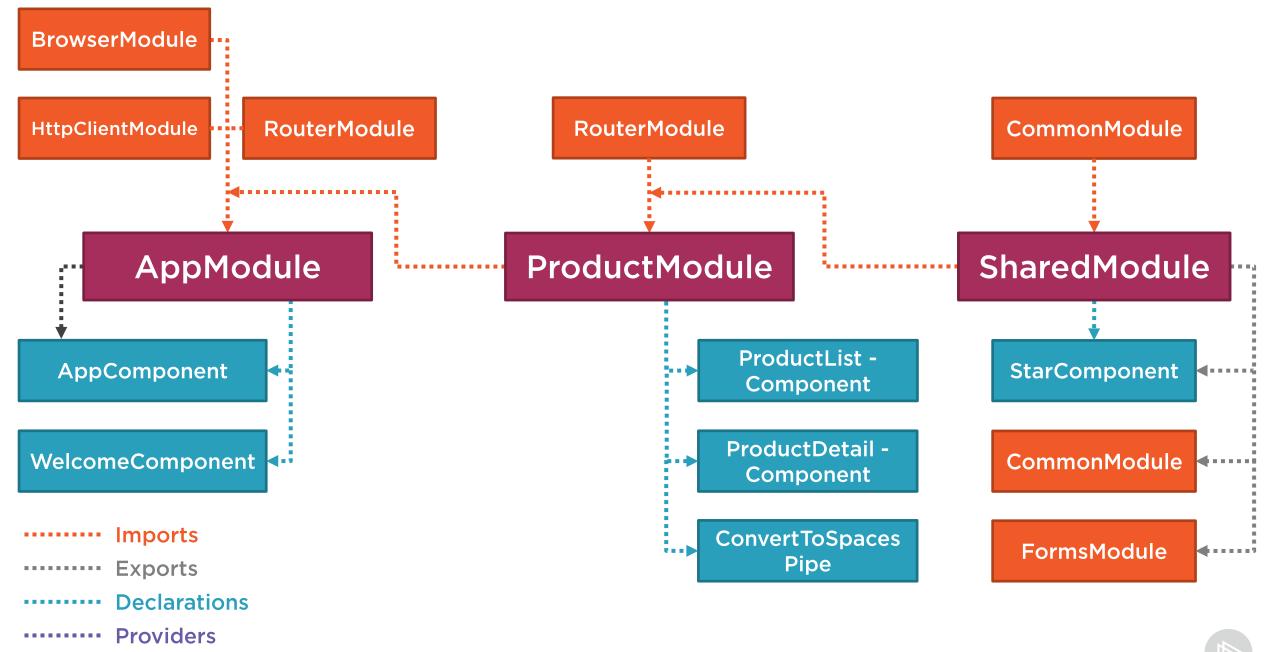


What Is an Angular Module?
Angular Module Metadata
Creating a Feature Module
Defining a Shared Module
Revisiting AppModule





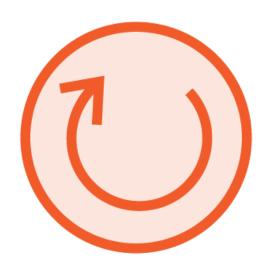




Bootstrap



# Up Next



**Angular CLI** 

