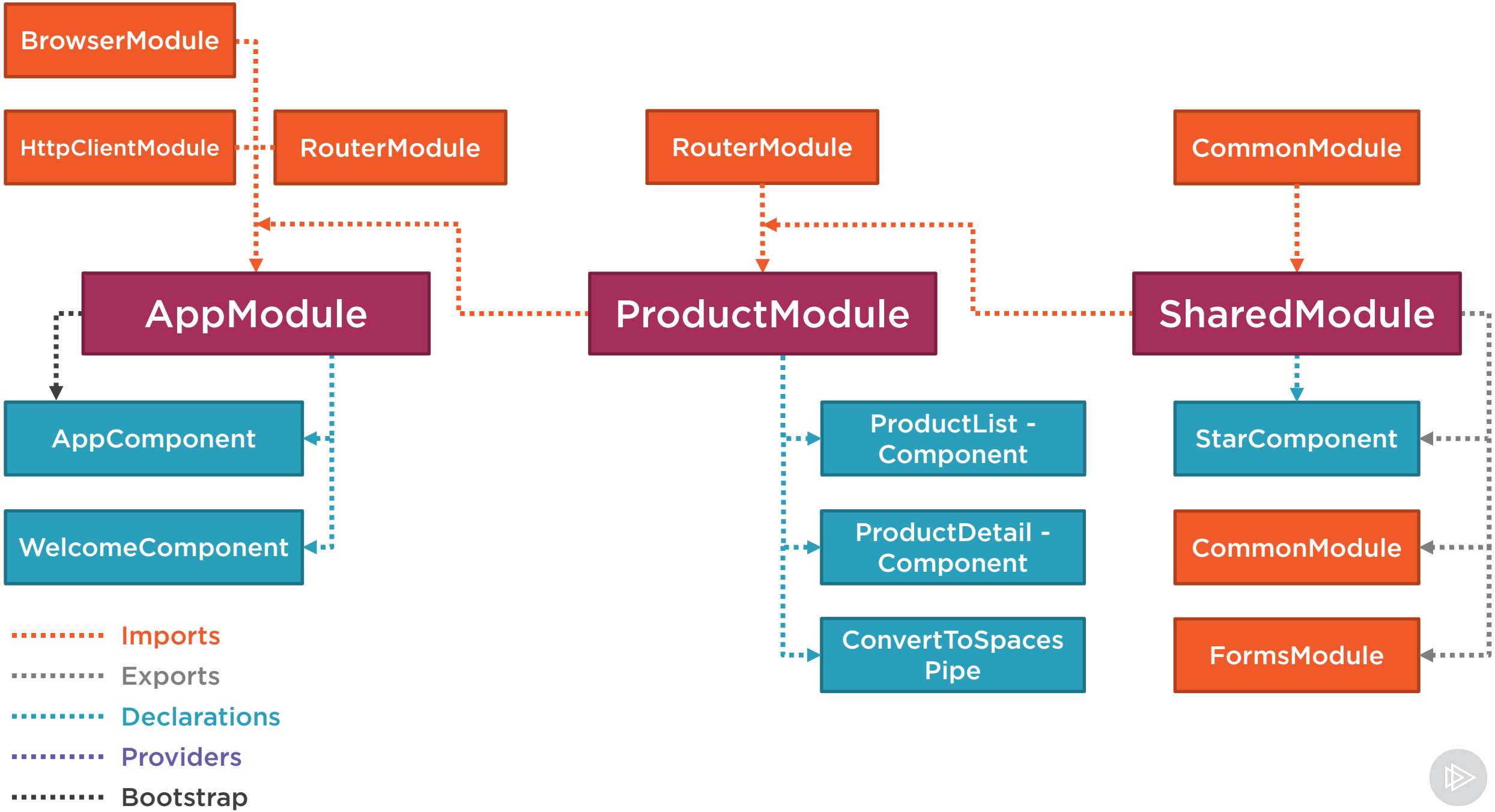
Angular Modules

Deborah Kurata

CONSULTANT | SPEAKER | AUTHOR | MVP | GDE

@deborahkurata | blogs.msmvps.com/deborahk/


BrowserModule
HttpClientModule
RouterModule
RouterModule
CommonModule
AppModule
ProductModule
SharedModule
AppComponent
WelcomeComponent
ProductList - Component
ProductDetail - Component
ConvertToSpaces Pipe
StarComponent
CommonModule
FormsModule
Imports
Exports
Declarations
Providers
Bootstrap

Module Overview

What Is an Angular Module?

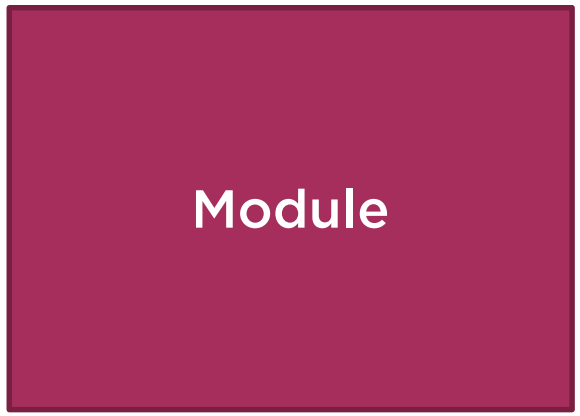
Angular Module Metadata

Creating a Feature Module

Defining a Shared Module

Revisiting AppModule

What Is an Angular Module?

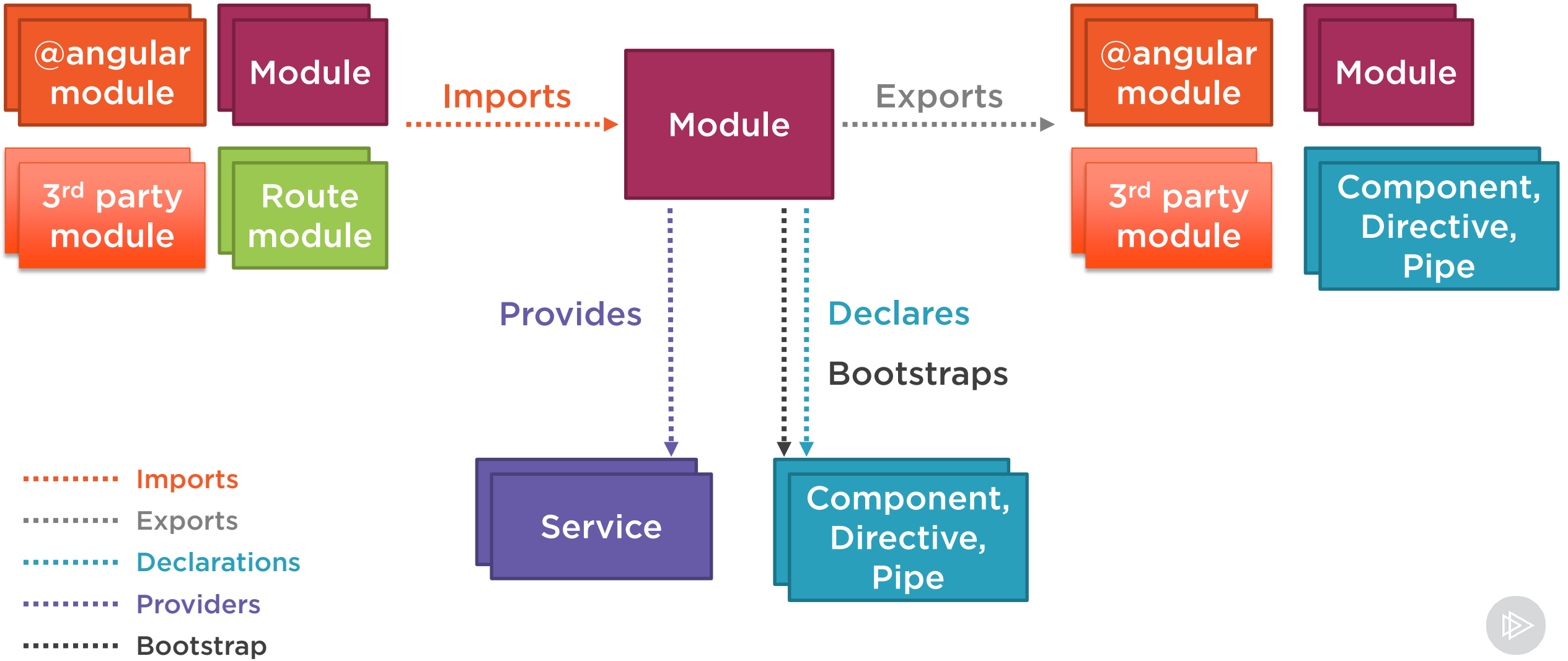


A class with an NgModule decorator

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

Angular Module



AppComponent

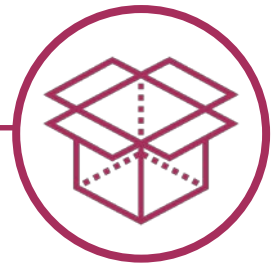
ProductList - Component

ProductDetail - Component

WelcomeComponent

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

RouterModule



Angular Module

AppComponent

ProductList - Component

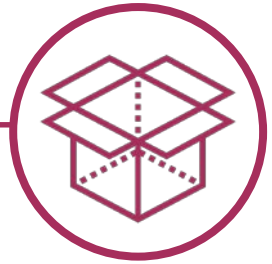
ProductDetail - Component

WelcomeComponent

RouterModule

sModule

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



Angular Module

AppComponent

ProductList - Component

ProductDetail - Component

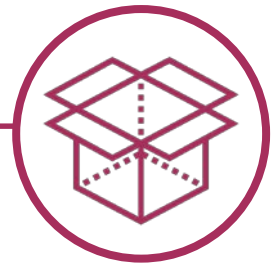
WelcomeComponent

```
...
<tr *ngFor='let product of products'>
  <td>
    <img *ngIf='showImage'
...
```

RouterModule

FormsModule

BrowserModule



Angular Module

AppComponent

ProductList - Component

ProductDetail - Component

WelcomeComponent

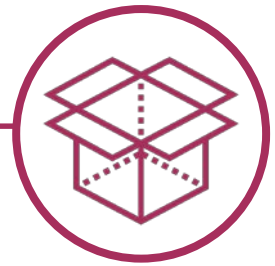
ConvertToSpaces Pipe

RouterModule

FormsModule

BrowserModule

```
...
<td>{{ product.productCode |
                convertToSpaces: '-' }}
</td>
...
```



Angular Module

AppComponent

ProductList - Component

ProductDetail - Component

WelcomeComponent

ConvertToSpaces Pipe

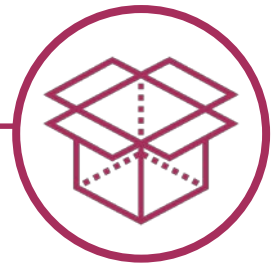
StarComponent

RouterModule

odule

Module

```
...
<pm-star
 [rating]='product.starRating'
 (ratingClicked)='onRatingClicked($event)'>
</pm-star>
...
```



Angular Module

AppComponent

ProductList - Component

ProductDetail - Component

WelcomeComponent

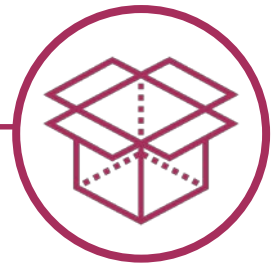
ConvertToSpaces Pipe

StarComponent

RouterModule

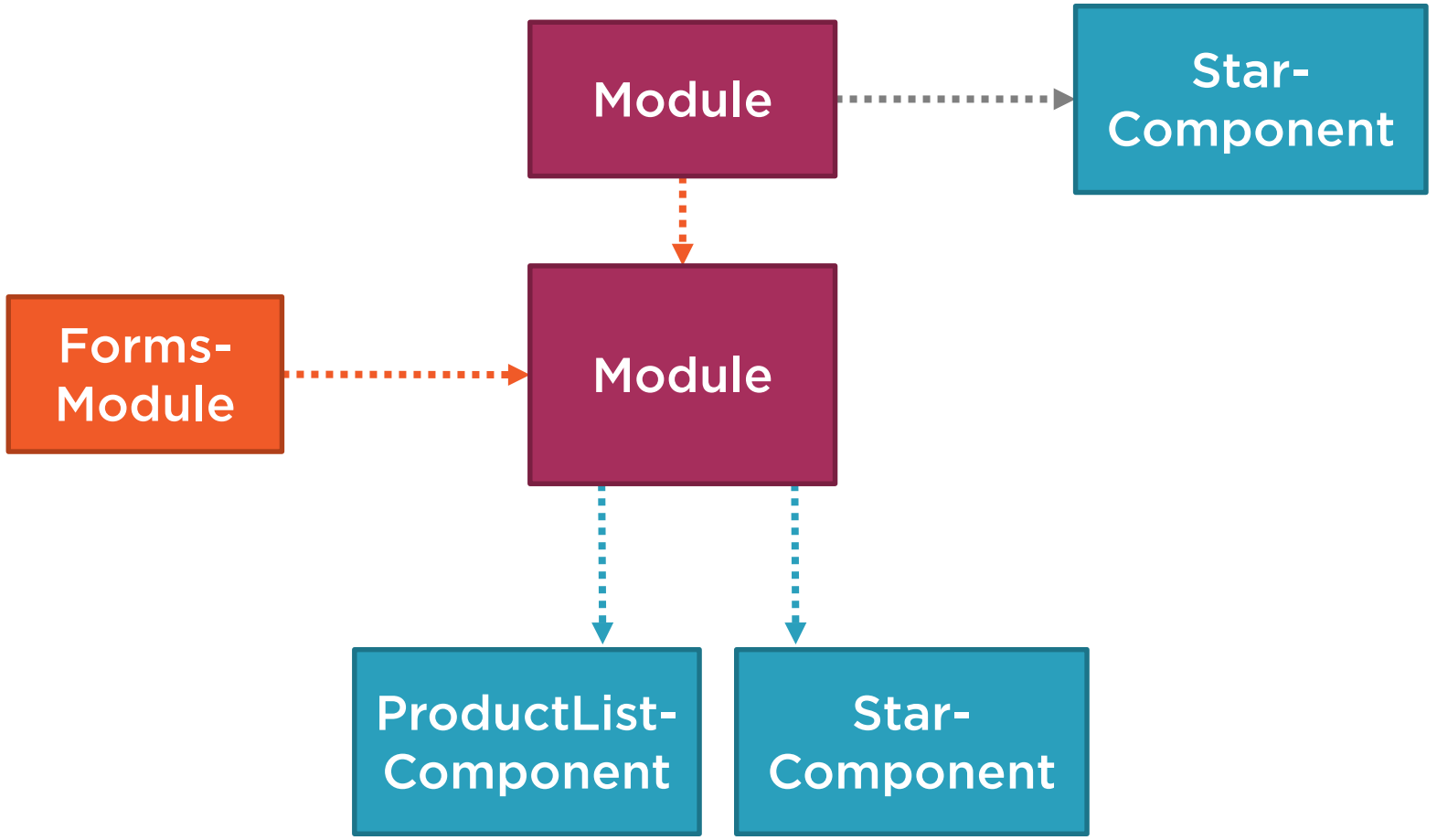
FormsModule

BrowserModule

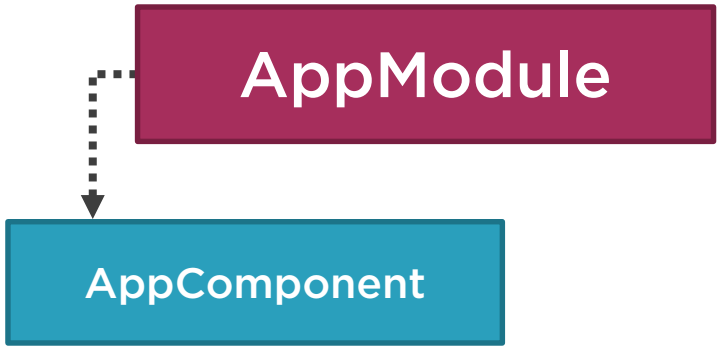


Angular Module

Template Resolution Environment



Bootstrap Array



app.module.ts

```
...
bootstrap: [ AppComponent ]
...
```

Bootstrap

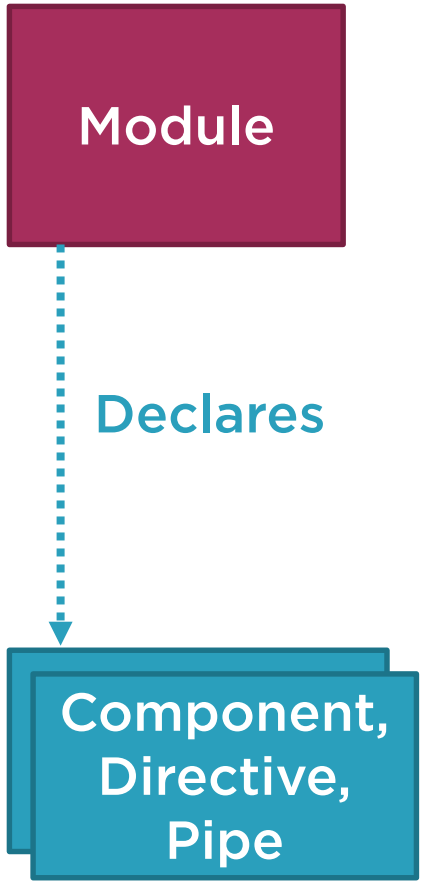
Bootstrap Array Truth #1

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

Declarations Array Truth #1

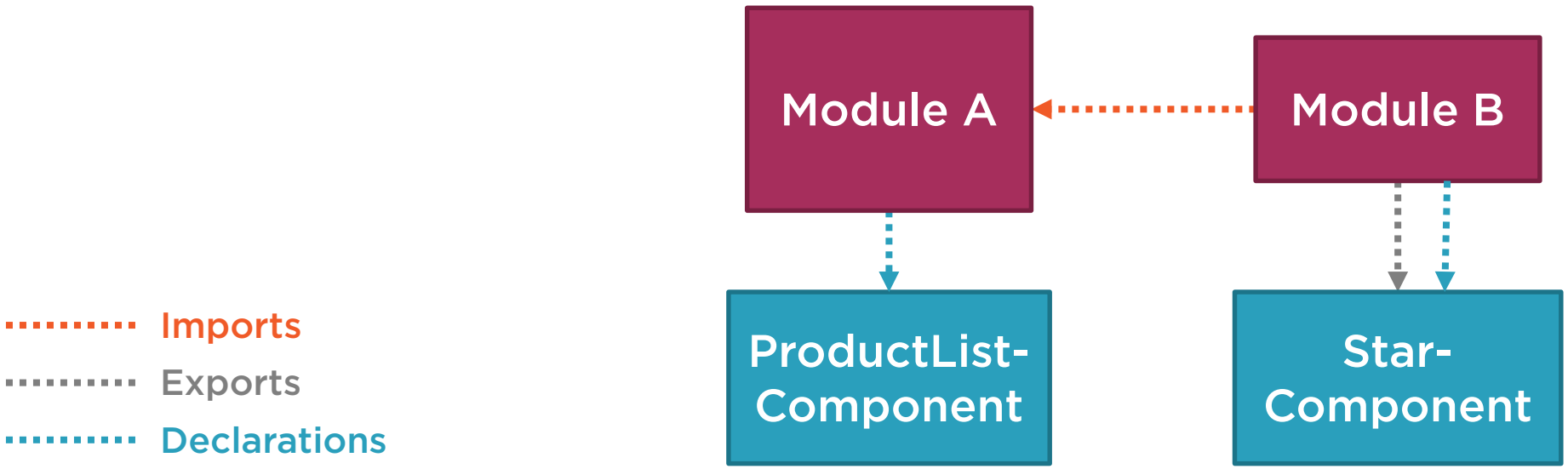
Every component, directive, and pipe we create must belong to one and only one Angular module.

Declarations Array Truth #2

Only declare components, directives and pipes.

Declarations Array Truth #3

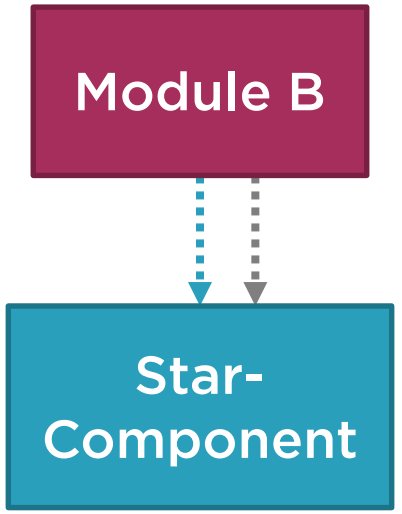
Never re-declare components, directives, or pipes that belong to another module



Declarations Array Truth #4

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.



Exports

Declarations

Declarations Array Truth #5

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

product-list.component.html

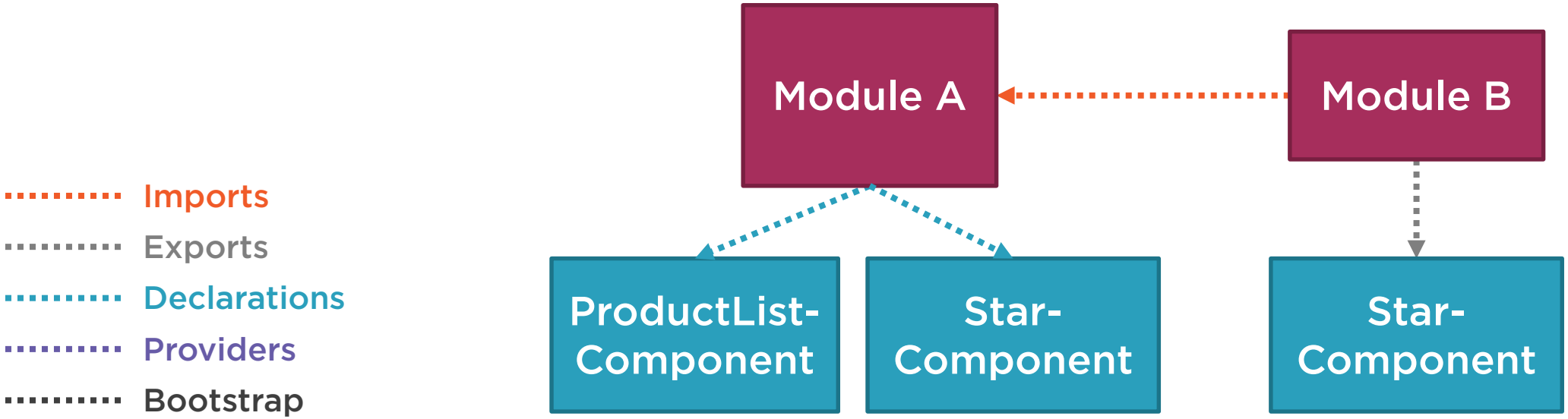
```
<pm-star ...>
</pm-star>
```

star.component.ts

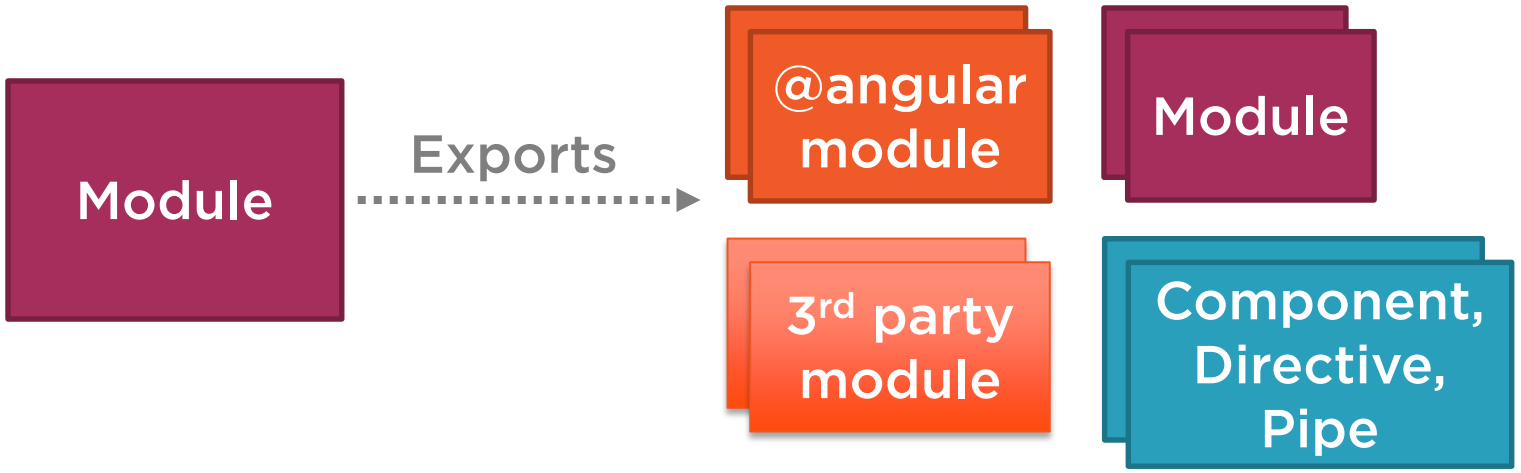
```
...
@Component({
    selector: 'pm-star',
    template: ...
})
...
```

Declarations Array Truth #5

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



Exports Array



Imports
Exports
Declarations
Providers
Bootstrap

Exports Array Truth #1

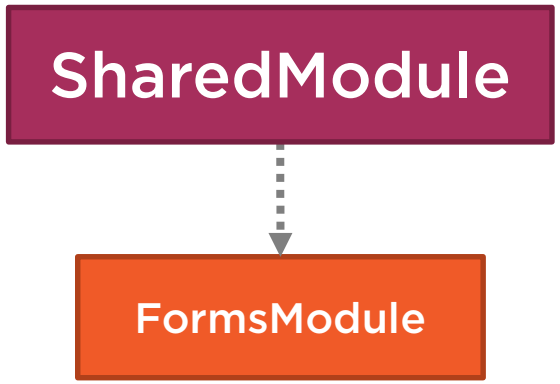
Export any component, directive, or pipe if other components need it.

Exports Array Truth #2

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

Exports Array Truth #3

We can re-export something without importing it first.

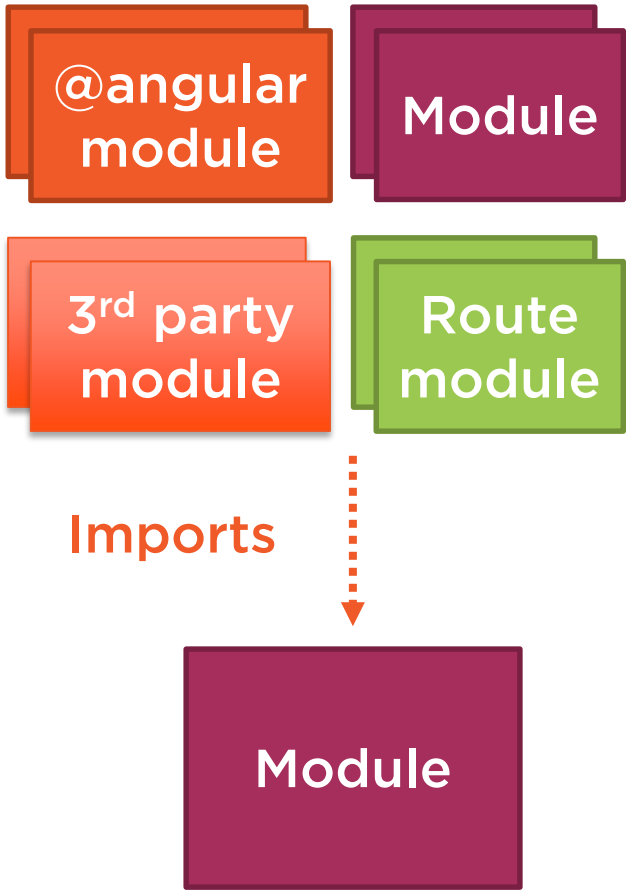


Imports
Exports
Declarations
Providers
Bootstrap

Exports Array Truth #4

Never export a service.

Imports Array

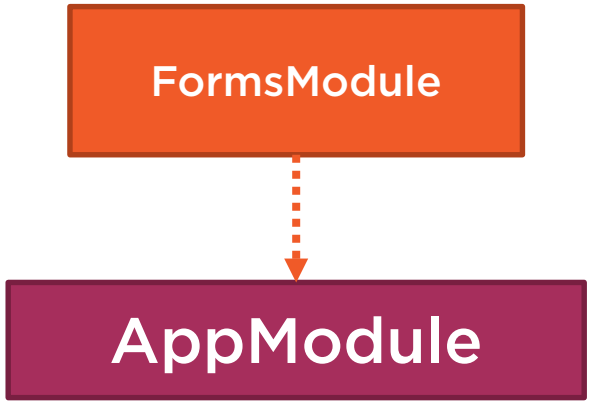


app.module.ts

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

Imports Array Truth #1

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



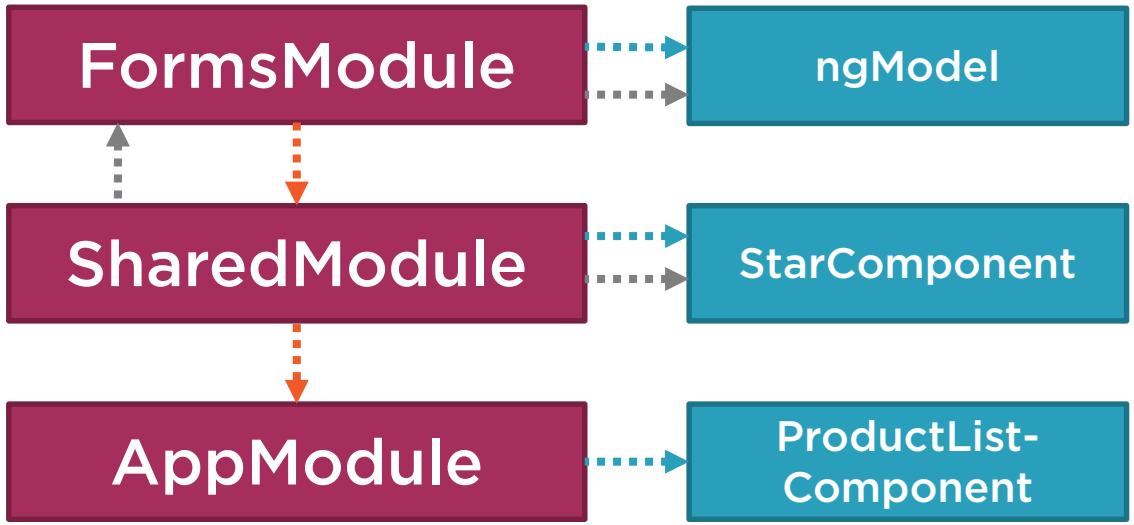
Imports
Exports
Declarations
Providers
Bootstrap

Imports Array Truth #2

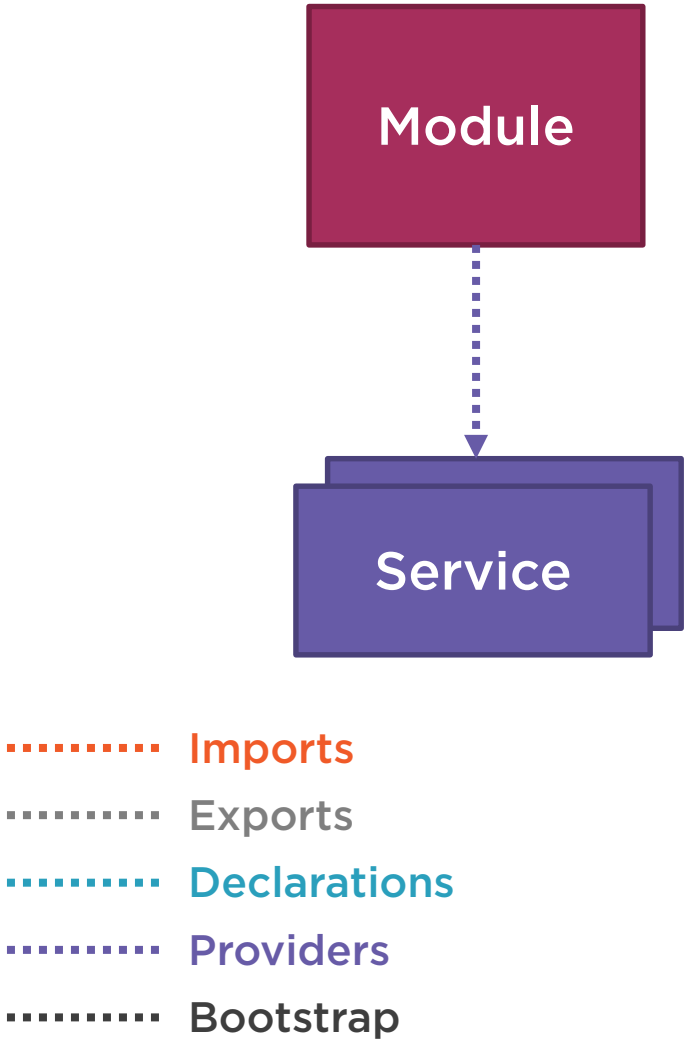
Only import what this module needs.

Imports Array Truth #3

Importing a module does NOT provide access to its imported modules



Providers Array



app.module.ts

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

product.service.ts

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

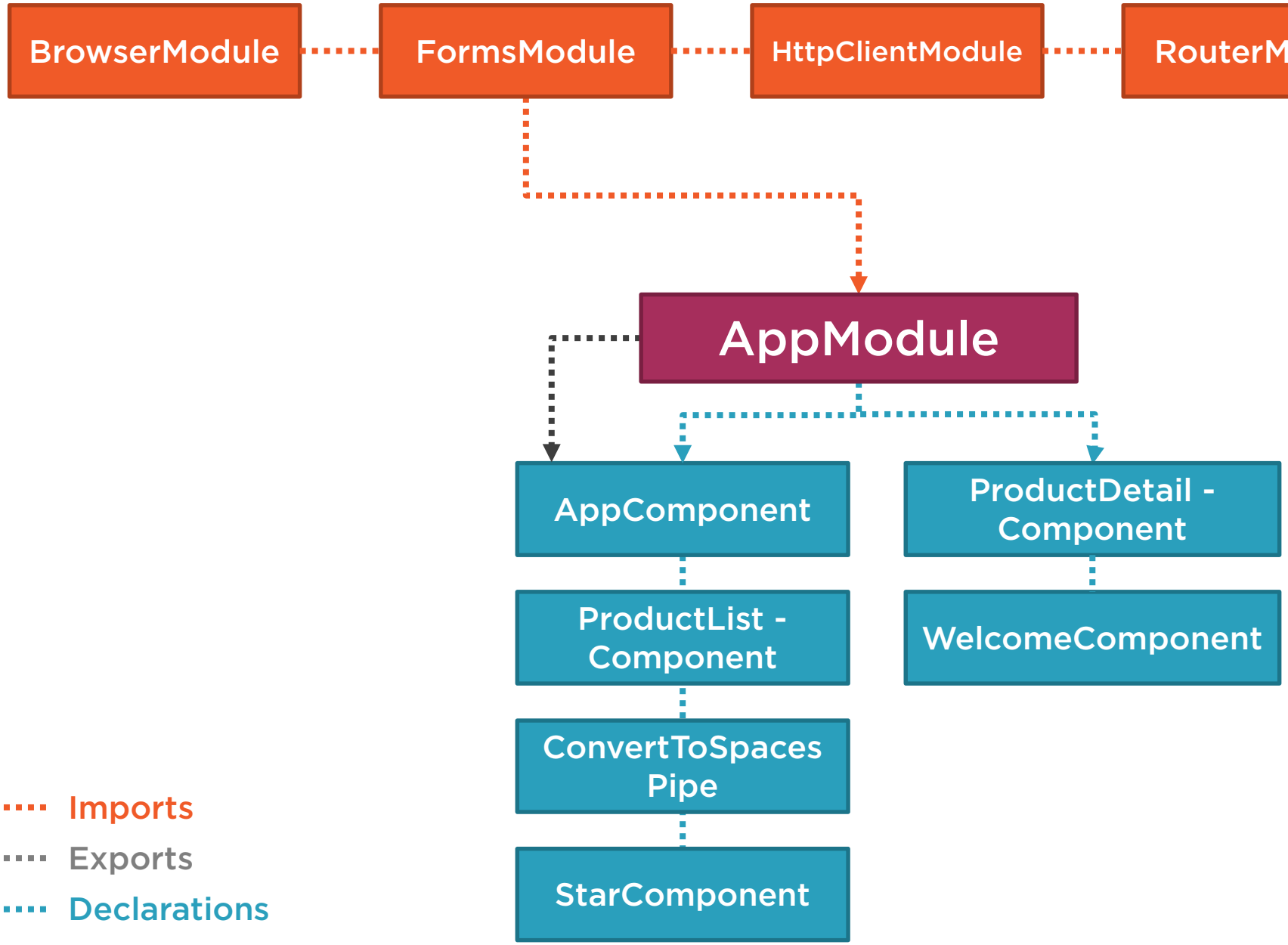
Providers Array Truth #1

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

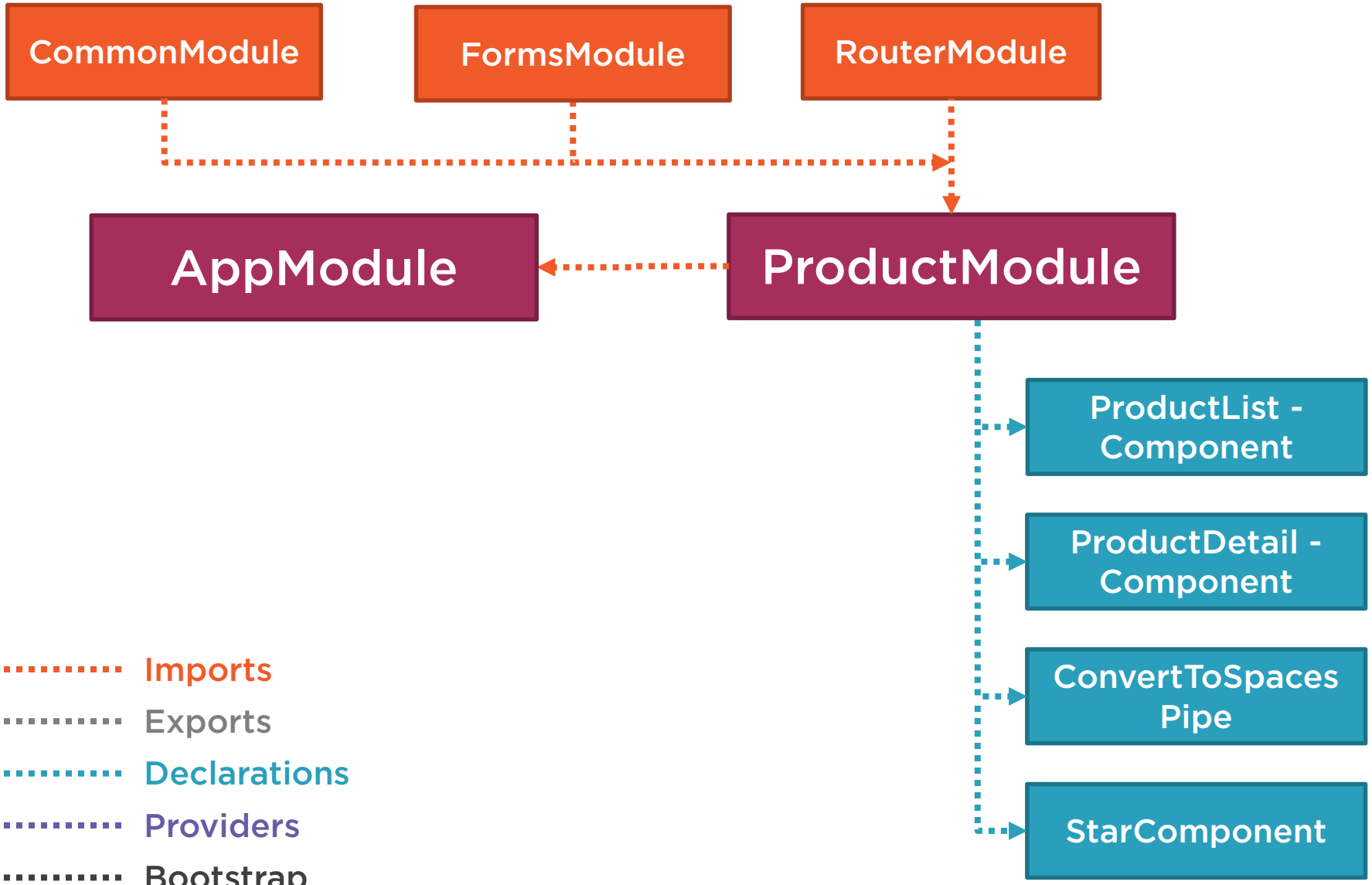
Providers Array Truth #2

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

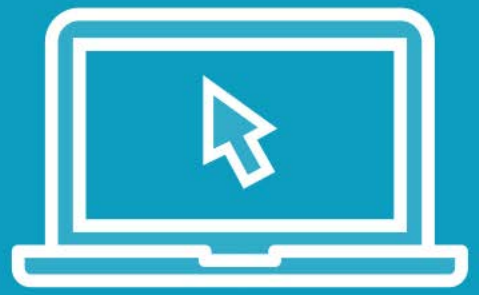
Consider building a CoreModule for services and importing it once in the AppModule.


BrowserModule
FormsModule
HttpClientModule
RouterModule
AppModule
AppComponent
ProductDetail - Component
ProductList - Component
WelcomeComponent
ConvertToSpaces Pipe
StarComponent
Imports
Exports
Declarations
Providers
Bootstrap

Defining a Feature Module

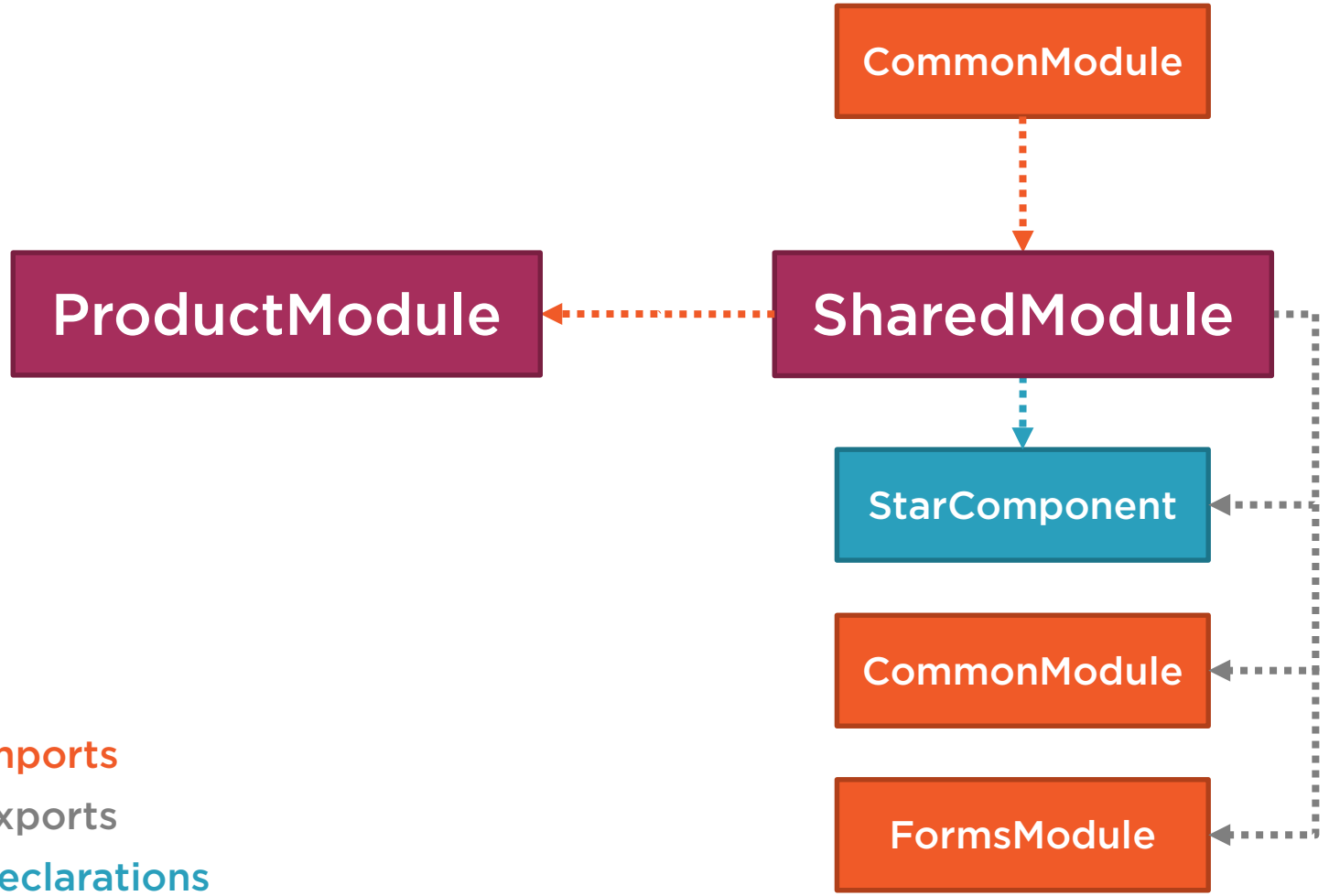


Demo

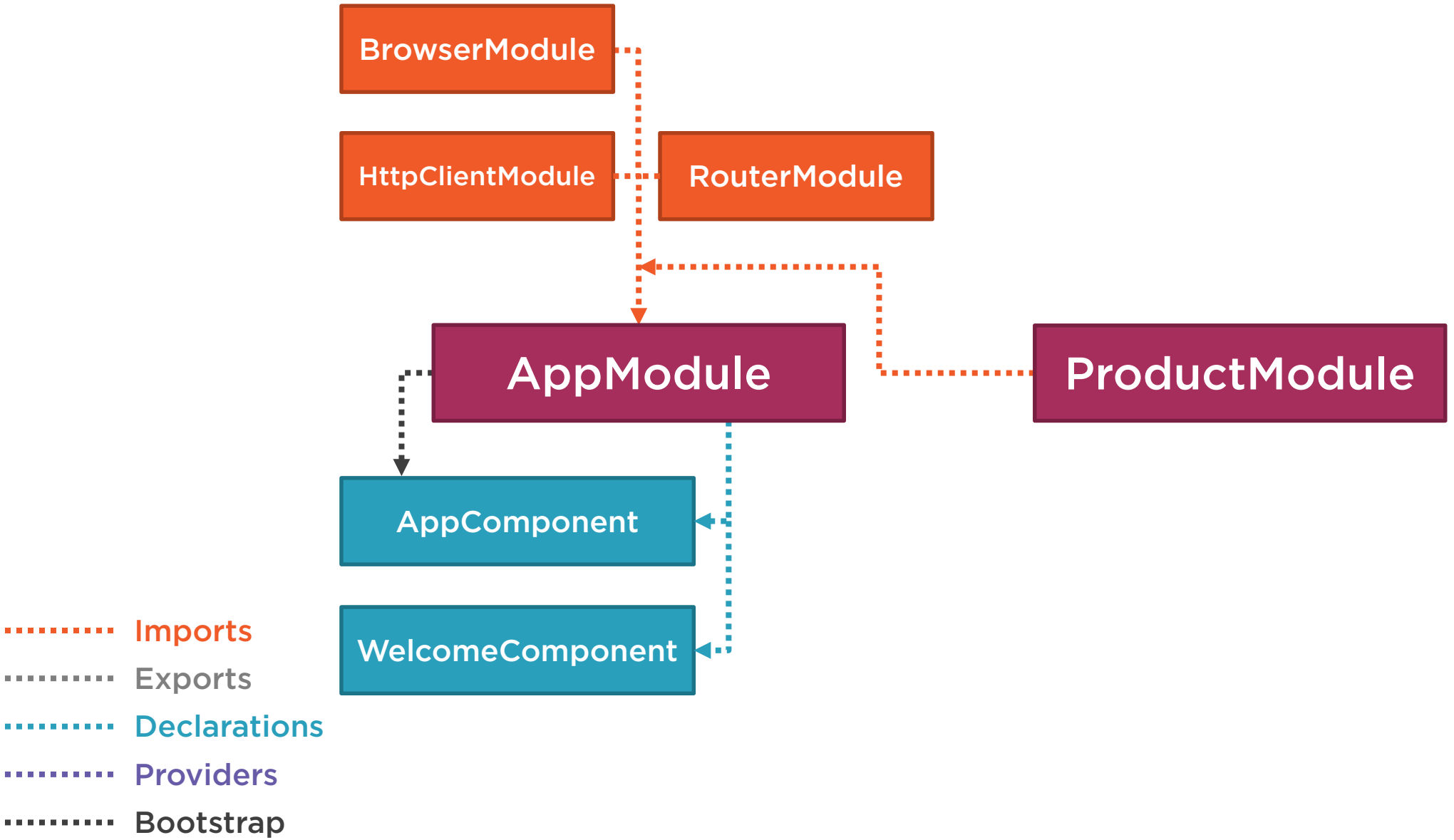


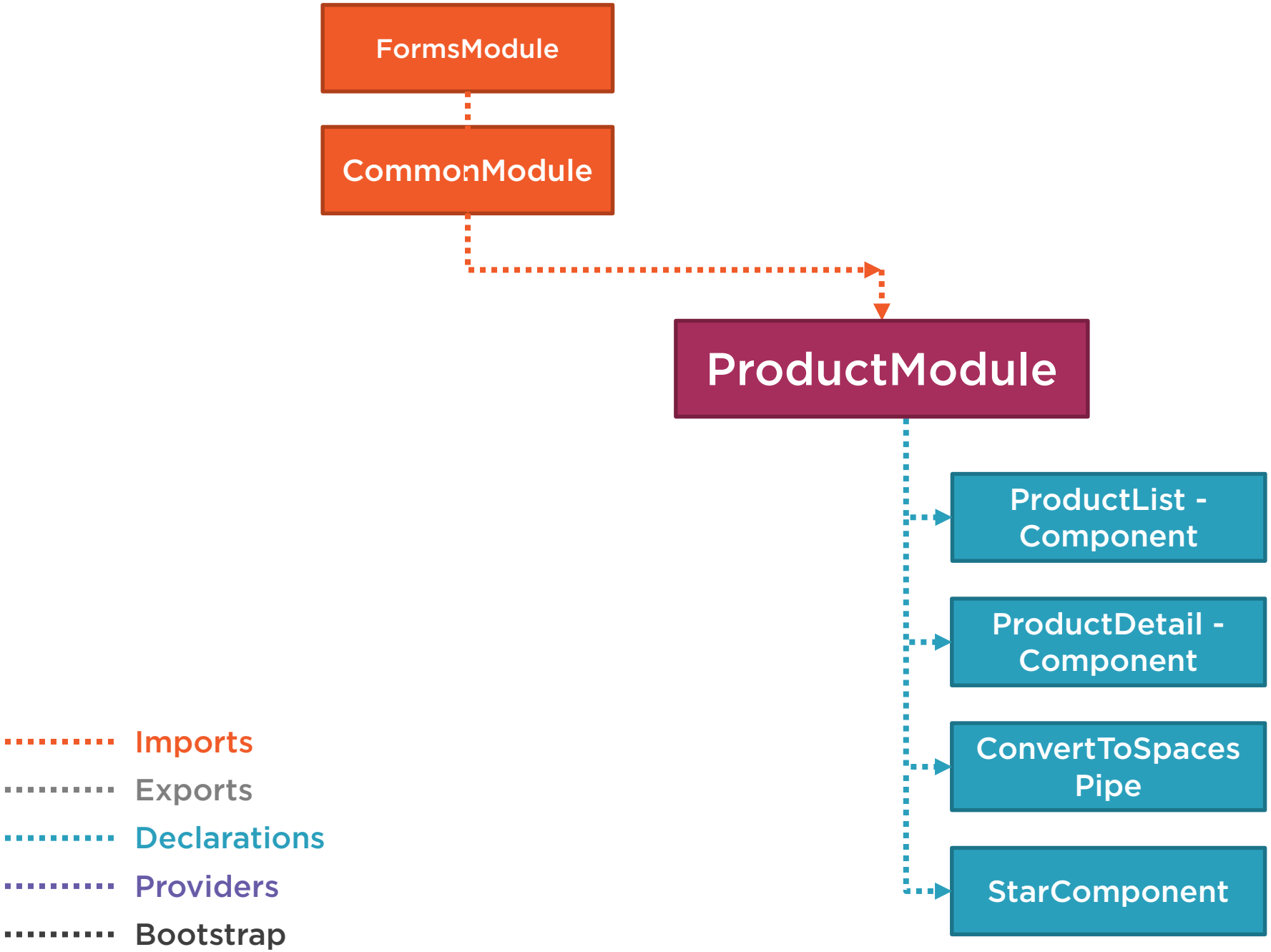
Building a Feature Module

Defining a Shared Module



Revisiting AppModule




FormsModule
CommonModule
ProductModule
ProductList - Component
ProductDetail - Component
ConvertToSpaces Pipe
StarComponent
Imports
Exports
Declarations
Providers
Bootstrap

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)
- Core module (CoreModule)
- Routing modules

Angular Module Checklist: NgModule Metadata

Bootstrap: Startup component(s)

Declarations: What belongs to this module

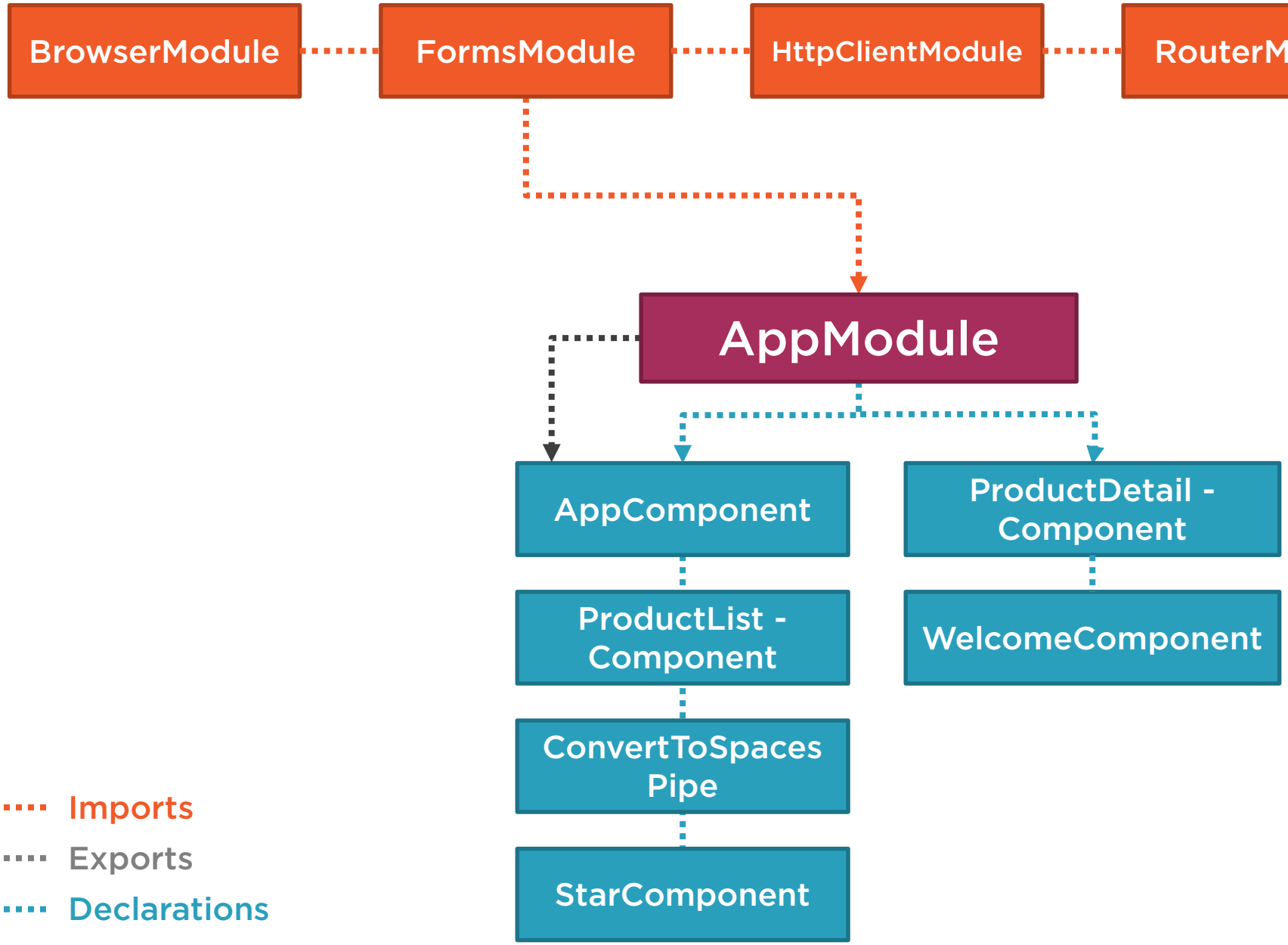
Exports: What an importing module can use

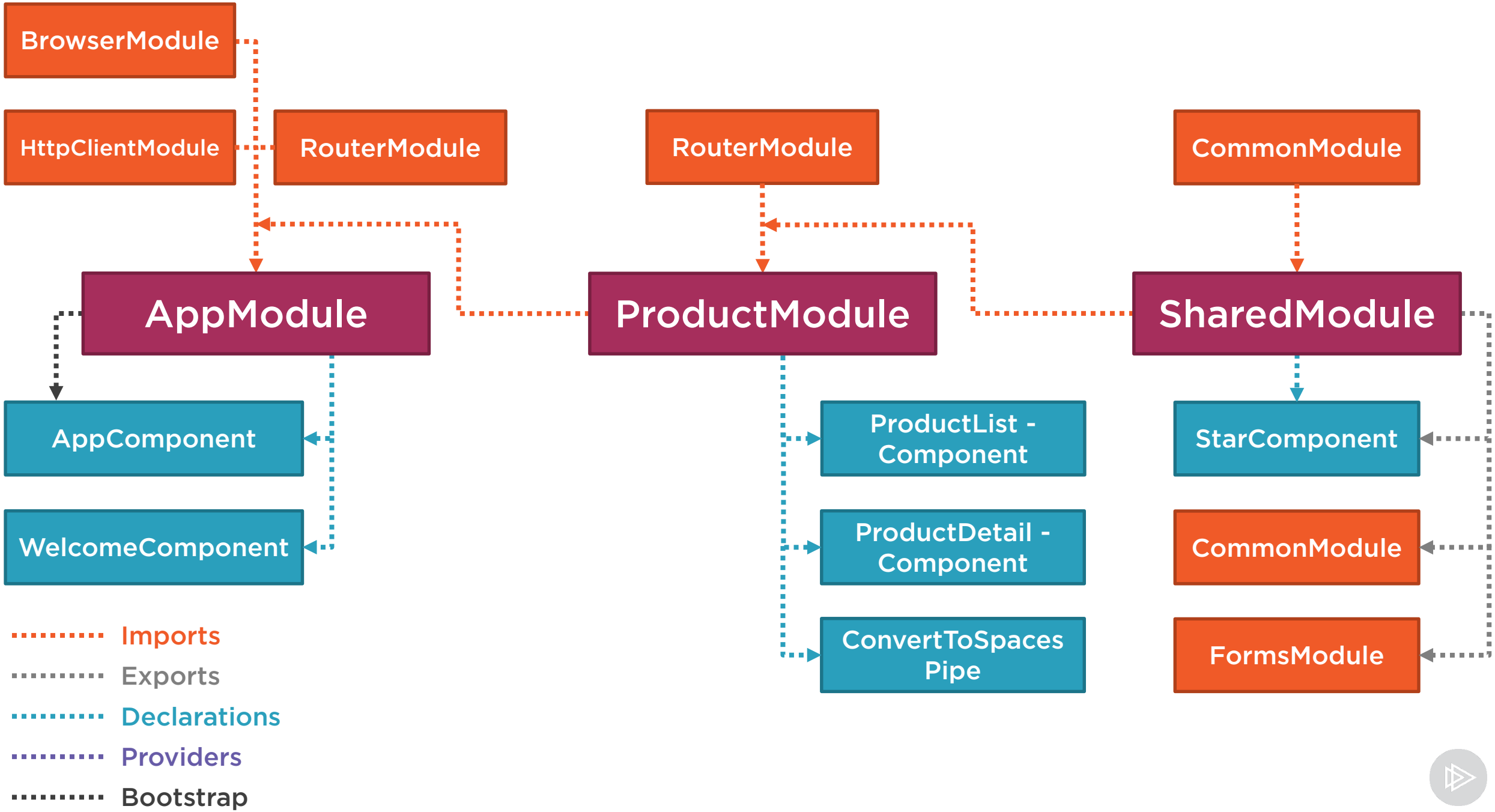
Imports: Supporting modules

Providers: Service providers

Summary

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


BrowserModule
FormsModule
HttpClientModule
RouterModule
AppModule
AppComponent
ProductDetail - Component
ProductList - Component
WelcomeComponent
ConvertToSpaces Pipe
StarComponent
Imports
Exports
Declarations
Providers
Bootstrap


BrowserModule
HttpClientModule
RouterModule
RouterModule
CommonModule
AppModule
ProductModule
SharedModule
AppComponent
WelcomeComponent
ProductList - Component
ProductDetail - Component
ConvertToSpaces Pipe
StarComponent
CommonModule
FormsModule
Imports
Exports
Declarations
Providers
Bootstrap

Up Next

Angular CLI