Angular: AppModule vs Standalone Applications (v15+)

1. Overview

Angular before v15 used the NgModule system for bootstrapping and organizing apps.

From Angular 15+, we can build Standalone Applications that don't need AppModule.

2. AppModule-based (Angular <= v14)

---

app.module.ts

```
import { NgModule } from '@angular/core';
import { BrowserModule } from '@angular/platform-browser';
import { RouterModule } from '@angular/router';
import { AppComponent } from './app.component';

@NgModule({
  declarations: [AppComponent],
  imports: [BrowserModule, RouterModule.forRoot(routes)],
  bootstrap: [AppComponent]
})
export class AppModule {}
```

main.ts

```
platformBrowserDynamic()
  .bootstrapModule(AppModule)
  .catch(err => console.error(err));
```

3. Standalone-based (Angular >= v15)

---

app.component.ts

```
import { Component } from '@angular/core';
import { RouterOutlet, RouterLink } from '@angular/router';

@Component({
  selector: 'app-root',
  standalone: true,
  imports: [RouterOutlet, RouterLink],
  templateUrl: './app.component.html'
})
export class AppComponent {}
```

app.config.ts

```
import { ApplicationConfig } from '@angular/core';
import { provideRouter } from '@angular/router';
import { routes } from './app.routes';

export const appConfig: ApplicationConfig = {
  providers: [provideRouter(routes)]
};
```

main.ts

```
import { bootstrapApplication } from '@angular/platform-browser';
import { AppComponent } from './app/app.component';
import { appConfig } from './app/app.config';

bootstrapApplication(AppComponent, appConfig)
  .catch(err => console.error(err));
```

4. Key Differences

Feature	AppModule-Based	Standalone-Based

---|---|---

Entry Point | AppModule | AppComponent

Bootstrap Function | bootstrapModule() | bootstrapApplication()

Config File | app.module.ts | app.config.ts

Component Declaration | declarations[] | standalone: true

Imports Location | NgModule imports[] | Component imports[]

Global Providers | providers[] in AppModule | providers[] in app.config.ts

Routing Setup | RouterModule.forRoot() | provideRouter()

HTTP Setup | HttpClientModule | provideHttpClient()

Backward Compatible | Yes | Yes

5. Benefits of Standalone

- Simpler project structure (no modules required)
- Faster startup and smaller bundles
- Easier to reason about dependencies
- Each component self-contained
- Works with old NgModules using importProvidersFrom()

6. Migration Note

Angular supports both approaches. You can migrate gradually:

- Mark components as standalone: true
- Replace RouterModule.forRoot() with provideRouter()
- Move providers to app.config.ts