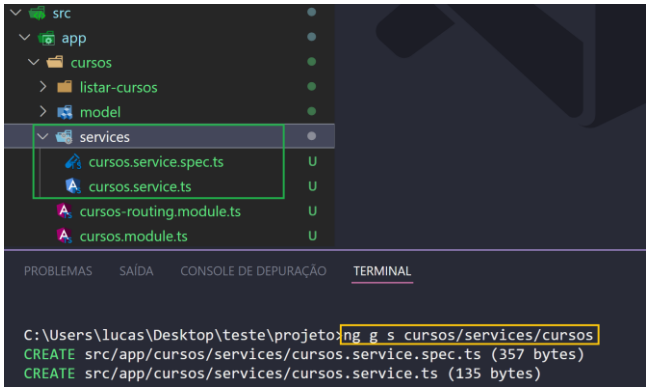
Serviço

A **camada de serviço** é responsável pela lógica de consumo de uma API *(ou qualquer outra lógica)*, deixando pro **componente** apenas **renderizar o que a camada de serviço tratou**.

Criar um serviço no **Angular** pode ser feito pela linha de comando com ***'ng g s xxxxx'***, é comum um módulo ter seus services:

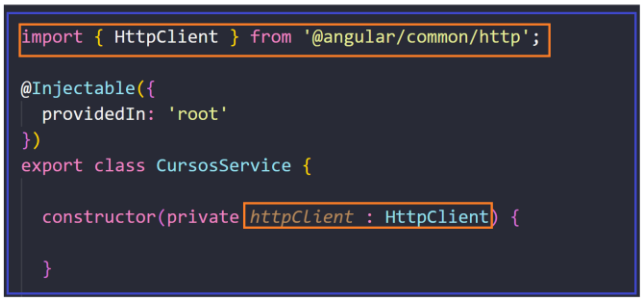


Para usar um **Service** em algum **componente** precisamos injeta-lo. Ou seja, no **construtor do componente** eu **passo a classe de Serviço**, dessa maneira eu posso **consumir as funções do serviço dentro do componente:**

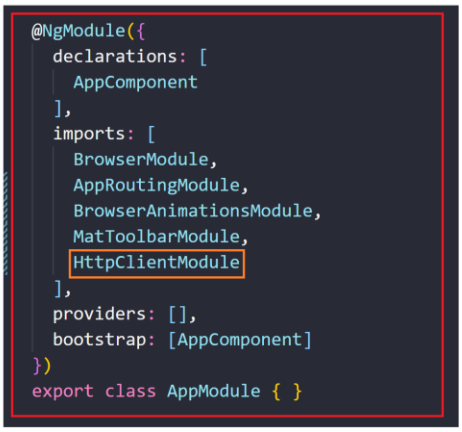
```
export class ListarCursosComponent implements OnInit {

  cursos : Curso[] =[];

  colunasExibidas =['nome', 'categoria'];

  constructor(serviceCurso : CursosService) {
    this.cursos = serviceCurso.listar();
  }

  ngOnInit(): void {
  }
```

Consumir métodos HTTP

O **Angular** tem uma classe utilitária que **facilita o consumo de API's,** essa classe é o **HttpClient.** Além de ser **importado na classe de serviço**, também **precisamos importa-lo no app.module**, para que ele fique disponível de maneira global no projeto *(pra usarmos em qualquer lugar)*:

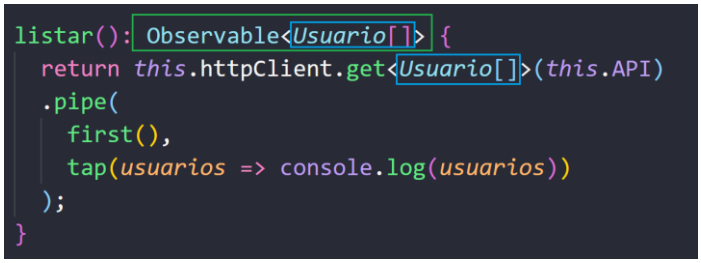


```
import { HttpClient } from '@angular/common/http';

@Injectable({
  providedIn: 'root'
})
export class CursosService {

  constructor(private httpClient : HttpClient) {

  }
```



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

O consumo no **Angular** acontece através do **Observable**, pensa no **Observable** como um ***"envelope"*** que c**ontém os dados da requisição**. Por exemplo, para buscar uma **lista de Usuários**, temos que **envelopar essa lista dentro do Observable:**



```
listar(): Observable<Usuario[]> {
  return this.httpClient.get<Usuario[]>(this.API)
  .pipe(
    first(),
    tap(usuarios => console.log(usuarios))
  );
}
```

Essas **funções dentro de pipe são do rxJs do Angular**, basicamente s**ão ações que você pode executar antes de devolver as respostas das requisições.**

Nesse caso ele ta usando first() para consumir a requisição e já encerrar ela, o tap() ta sendo usado para imprimir a resposta da requisição no console para ajudar no debug.