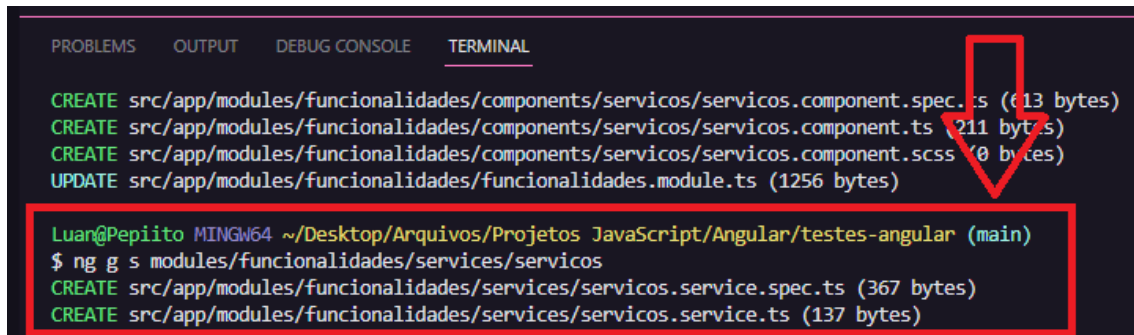


## Serviços

Os serviços são responsáveis por fazer o tratamento de dados e entregá-los já prontos para que o componente disponibilize para renderização.

Para utilizar:

1) Primeiro precisamos criar o serviço utilizando o comando “ng generate service caminhoDoArquivo/nomeDoArquivo” ou “ng g s caminhoDoArquivo/nomeDoArquivo”. Serão gerados os arquivos de serviço e de teste do serviço.

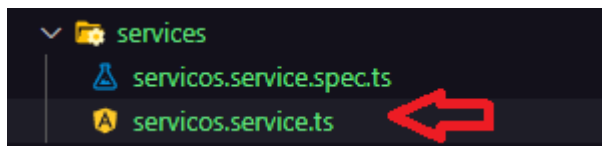


```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

CREATE src/app/modules/funcionalidades/components/servicos/servicos.component.spec.ts (613 bytes)
CREATE src/app/modules/funcionalidades/components/servicos/servicos.component.ts (211 bytes)
CREATE src/app/modules/funcionalidades/components/servicos/servicos.component.scss (0 bytes)
UPDATE src/app/modules/funcionalidades/funcionalidades.module.ts (1256 bytes)

Luan@Pepiito MINGW64 ~/Desktop/Arquivos/Projetos JavaScript/Angular/testes-angular (main)
$ ng g s modules/funcionalidades/services/servicos
CREATE src/app/modules/funcionalidades/services/servicos.service.spec.ts (367 bytes)
CREATE src/app/modules/funcionalidades/services/servicos.service.ts (137 bytes)
```

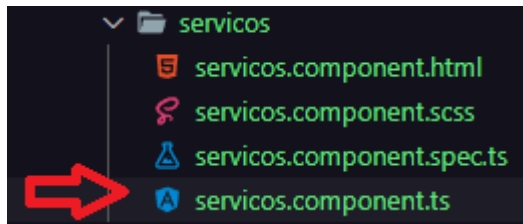
2) Agora podemos abrir o arquivo “.service.ts” e realizar as configurações necessárias. Neste exemplo, nosso serviço será responsável por receber um array com valores e disponibilizar um método que retorne este array. Vamos criar o array “nomes” e o método “getNomes”.



```
src > app > modules > funcionalidades > services > servicos.service.ts > ...




1  import { Injectable } from '@angular/core';
2
3  @Injectable({
4    providedIn: 'root'
5  })
6  export class ServicosService {
7
8    nomes: string[] = ['João', 'Maria', 'José']
9
10   constructor() { }
11
12   getNomes(): string[] {
13     return this.nomes
14   }
15 }
16
```

3) Agora podemos configurar o arquivo “.ts” do componente para receber os valores disponibilizados pelo service. Para isso, no arquivo “.ts” do componente, vamos criar um array chamado “lista” que será consultado pelo arquivo html para renderizar as informações no navegador.





```
src > app > modules > funcionalidades > componentes > servicos > servicos.component.ts > ...
1  import { Component, OnInit } from '@angular/core';
2  import { ServicosService } from '../services/servicos.service';
3
4  @Component({
5    selector: 'app-servicos',
6    templateUrl: './servicos.component.html',
7    styleUrls: ['./servicos.component.scss']
8  })
9  export class ServicosComponent implements OnInit {
10
11    lista: string[] = []
12
13    endereco: string = '../../../assets/Servicos.pdf'
14    github: string = 'https://github.com/LuanSOliveira/Testes-Angular/t
15
16    constructor(private servicoService: ServicosService) {}
17
18    ngOnInit(): void {
19      this.lista = this.servicoService.getNomes()
20    }
21
22  }
23
```

4) Agora, dentro do construtor do componente, podemos importar o service criado. Para isso utilizamos a seguinte expressão “private X: nomeDoServiço”. Assim, nosso componente terá acesso aos métodos criados no serviço.

```
src > app > modules > funcionalidades > components > servicos >  servicos.component.ts > ...
1  import { Component, OnInit } from '@angular/core';
2  import { ServicosService } from '../services/servicos.service'; 
3
4  @Component({
5    selector: 'app-servicos',
6    templateUrl: './servicos.component.html',
7    styleUrls: ['./servicos.component.scss']
8  })
9  export class ServicosComponent implements OnInit {
10
11    lista: string[] = []
12
13    endereco: string = '../assets/Servicos.pdf'
14    github: string = 'https://github.com/LuanSoliveira/Testes-Angular/tree/main/src/'
15
16    constructor(private servicoService: ServicosService) {} 
17
18    ngOnInit(): void {
19      this.lista = this.servicoService.getNomes()
20    }
21
22  }
23
```

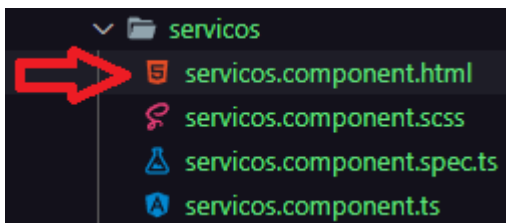
5) Para este exemplo vamos atribuir o retorno do método “getNomes” do service ao array “lista”. Para que isso ocorra logo que o componente for renderizado, vamos utilizar o método “OnInit” do angular/core. Vamos realizar o import do “OnInit” e implementa-lo na classe.

```
src > app > modules > funcionalidades > components > servicos >  servicos.component.ts > ...
1  import { Component, OnInit } from '@angular/core';
2  import { ServicosService } from '../services/servicos.service';
3
4  @Component({
5    selector: 'app-servicos',
6    templateUrl: './servicos.component.html',
7    styleUrls: ['./servicos.component.scss']
8  })
9  export class ServicosComponent implements OnInit { 
10
11    lista: string[] = []
12
13    endereco: string = '../assets/Servicos.pdf'
14    github: string = 'https://github.com/LuanSoliveira/Testes-Angular/tree/main/src/'
15
16    constructor(private servicoService: ServicosService) {}
17
18    ngOnInit(): void {
19      this.lista = this.servicoService.getNomes()
20    }
21
22  }
23
```

6) Agora podemos fazer a chamada do método com a função “ngOnInit” e dentro dele informar o que deve ser executado quando o componente for inicializado. No exemplo estamos atribuindo em array o valor do método “getNomes” do serviço importado.

```
src > app > modules > funcionalidades > components > servicos > servicos.component.ts > ...
1  import { Component, OnInit } from '@angular/core';
2  import { ServicosService } from '../services/servicos.service';
3
4  @Component({
5    selector: 'app-servicos',
6    templateUrl: './servicos.component.html',
7    styleUrls: ['./servicos.component.scss']
8  })
9  export class ServicosComponent implements OnInit {
10
11    lista: string[] = []
12
13    endereco: string = '../assets/Servicos.pdf'
14    github: string = 'https://github.com/LuanSoliveira/Testes-Angular/'
15
16    constructor(private servicoService: ServicosService) {}
17
18    ngOnInit(): void {
19      this.lista = this.servicoService.getNomes()
20    }
21
22  }
23
```

7) Agora podemos atribuir os valores do array lista no arquivo html do componente.



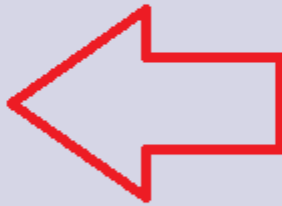
```
src > app > modules > funcionalidades > components > servicos > servicos.component.html > div.div-compo
Go to component
1  <div class="div-componente">
2    <h2 class="title-componente">Serviços</h2>
3    <div class="componente-container">
4      <p class="descricao-componente">
5        Os serviços são responsáveis por fazer o tratamento de dados e entreg
6        Neste exemplo, os valores dos nomes estão sendo tratados pelo arquivo
7        para que sejam atribuídos em variáveis e renderizados no html.
8      </p>
9    </div>
10   <div class="componente-exemplo">
11     <ul class="exemplo-lista">
12       <li class="exemplo-lista-item">Nome: {{lista[0]}}</li>
13       <li class="exemplo-lista-item">Nome: {{lista[1]}}</li>
14       <li class="exemplo-lista-item">Nome: {{lista[2]}}</li>
15     </ul>
16   </div>
17   <div class="componente-buttons-container">
18     <app-arquivo-pdf [endereco]="endereco"></app-arquivo-pdf>
19     <app-repositorio [github]="github"></app-repositorio>
20   </div>
21 </div>
```

8) Desta forma, serão renderizados no navegador os valores definidos no serviço criado.

## Serviços

Os serviços são responsáveis por fazer o tr  
valores dos nomes estão sendo tratados pe  
variáveis e renderizados no html.

Nome: João  
Nome: Maria  
Nome: José



**Como Utilizar**

**Repositório**