UNIVERSIDADE FEDERAL DO CEARÁ
CAMPUS QUIXADÁ

Qualidade de Software 2025.2

Refatoração de Code Smells em Frameworks Front-end

Taiga-family/Taiga-ui

Camile Isidorio Araujo
Francisco Werley da Silva

TAIGA-UI

Taiga UI é uma biblioteca de componentes de interface para aplicações Angular.

- Oferece componentes prontos e reutilizáveis (botões, formulários, tabelas, modais, layouts, etc.)
- É modular e performática, permitindo importar apenas o que é necessário
- Possui alto nível de customização, com suporte a temas e CSS custom properties
- Segue boas práticas do Angular e é usada em projetos reais
- Open source (Apache 2.0), podendo ser usada em projetos pessoais e comerciais

Tipos de Code Smells

Tipo	Descrição
DOM	Dependency on Module - componente depende diretamente de muitos módulos concretos, em vez de depender de abstrações.
LC	Large Component - Componente com muitas responsabilidades.
IIC	Inheritance Instead of Composition - Uso de herança onde composição seria mais adequada.
TMI	Too Many Inputs - Componente recebe dados em excesso.
ANY	Overusing Any Type - Uso excessivo do tipo any, perdendo segurança de tipos.
LF	Large File - Arquivo grande e difícil de manter.

Usando a ferramenta

- Executando a ferramenta ***angular-code-smells***.
- 331 arquivos identificaram algum tipo de smell.

```
C:\Users\camil\angular-code-smells
```

(index)	file	DOM	IIC	LC	ANY	TMI	LF	EPC
0	'..\\taiga-ui\\projects\\addon-charts\\components\\axes\\axes.component.ts'	0	0	1	0	0	0	0
1	'..\\taiga-ui\\projects\\addon-charts\\components\\line-chart\\line-chart.component.ts'	0	0	1	0	0	0	0
2	'..\\taiga-ui\\projects\\addon-charts\\components\\line-days-chart\\line-days-chart.component.ts'	0	0	1	0	0	0	0
3	'..\\taiga-ui\\projects\\addon-commerce\\components\\input-card-group\\input-card-group.component.ts'	1	0	1	0	0	1	0
4	'..\\taiga-ui\\projects\\addon-commerce\\types\\currency-code.ts'	0	0	0	0	0	1	0
327	'..\\taiga-ui\\projects\\testing\\utils\\active-element.ts'	1	0	0	0	0	0	0
328	'..\\taiga-ui\\projects\\testing\\utils\\input\\cleaner.unit-common.ts'	1	0	0	0	0	0	0
329	'..\\taiga-ui\\projects\\testing\\utils\\input\\filler.unit-common.ts'	1	0	0	0	0	0	0
330	'..\\taiga-ui\\projects\\testing\\utils\\native-input.page-object.ts'	1	0	0	0	0	0	0
331	'..\\taiga-ui\\projects\\testing\\utils\\page-object.ts'	1	0	0	0	0	0	0

Code smells encontrados

- No total foram encontrados 365 code smells, abaixo está a quantidade por cada tipo:

DOM	IIC	LC	ANY	TMI	LF
74	60	117	11	1	102

Refatoração

- Cada um ficou com 20 smells para refatorar.
- Os tipos escolhidos foram:

ANY	LF	LC	IIC
10	10	10	10

Refatoração - LF

- **Problema**:
  - Função/Método muito grande, com várias responsabilidades misturadas, difícil de entender e manter.
- **Solução**:
  - Dividir a função em métodos menores, cada um com uma única responsabilidade clara.

Refatoração - LF

- **Antes**:

- Apende é responsável por calcular uma nova data somando anos, dias e meses.
- Toda lógica concentrada em um único método

```
public override append({year = 0, month = 0, day = 0}: TuiDayLike): TuiDay {
    const totalMonths = (this.year + year) * MONTHS_IN_YEAR + this.month + month;
    let years = Math.floor(totalMonths / MONTHS_IN_YEAR);
    let months = totalMonths % MONTHS_IN_YEAR;

    const monthDaysCount = TuiMonth.getMonthDaysCount(
        months,
        TuiYear.isLeapYear(years),
    );
    const currentMonthDaysCount = TuiMonth.getMonthDaysCount(
        this.month,
        TuiYear.isLeapYear(years),
    );
    let days = day;

    if (this.day >= monthDaysCount) {
        days += this.day - (currentMonthDaysCount - monthDaysCount);
    } else if (
        currentMonthDaysCount < monthDaysCount &&
        this.day === currentMonthDaysCount
    ) {
        days += this.day + (monthDaysCount - currentMonthDaysCount);
    } else {
        days += this.day;
    }

    while (days > TuiMonth.getMonthDaysCount(months, TuiYear.isLeapYear(years))) {
        days -= TuiMonth.getMonthDaysCount(months, TuiYear.isLeapYear(years));

        if (months === TuiMonthNumber.December) {
```

Refatoração - LF

- **Depois**:

```
public override append({year = 0, month = 0, day = 0}: TuiDayLike): TuiDay {
    const {years, months} = this.calculateInitialMonthsAndYears(year, month);
    const days = this.calculateInitialDays(day, years, months);
    const adjustedDaysOver = this.adjustDaysOverLimit(days, years, months);

    return this.adjustDaysUnderLimit(
        adjustedDaysOver.days,
        adjustedDaysOver.years,
        adjustedDaysOver.months,
    );
}
```

Refatoração - LC

- **Problema**:
  - Classe/Componente muito grande, concentrando muitas responsabilidades e ficando difícil de entender, manter e testar.
- **Solução**:
  - Aplicar Extract Class, dividindo a classe em classes menores com responsabilidades bem definidas.

Refatoração - LC

- **Antes**:

- TuiAxes: responsável por organizar e exibir as configurações dos eixos de um gráfico.

- Inputs independentes espalhados pelo código

- Lógica acessava inputs diretamente

```
export const TUI_ALWAYS_DASHED: TuiLineHandler = (index) =>
    (index && 'dashed') || 'solid';
export const TUI_ALWAYS_DOTTED: TuiLineHandler = (index) =>
    (index && 'dotted') || 'solid';
export const TUI_ALWAYS_SOLID: TuiLineHandler = () => 'solid';
export const TUI_ALWAYS_NONE: TuiLineHandler = () => 'none';

@Component({
    selector: 'tui-axes',
    templateUrl: './axes.template.html',
    styleUrl: './axes.style.less',
    changeDetection: ChangeDetectionStrategy.OnPush,
    host: {
        dir: 'ltr',
        '[class._centered]': 'centeredXLabels()',
    },
})
export class TuiAxes {
    public readonly axisXLabels = input<ReadonlyArray<string | null>>([]);
    public readonly axisYInset = input(false);
    public readonly axisYLabels = input<readonly string[]>([]);
    public readonly axisYName = input('');
    public readonly axisYSecondaryInset = input(false);
    public readonly axisYSecondaryLabels = input<readonly string[]>([]);
    public readonly axisYSecondaryName = input('');
    public readonly centeredXLabels = input(false);
    public readonly horizontalLines = input(1);
    public readonly horizontalLinesHandler = input<TuiLineHandler>(TUI_ALWAYS_SOLID);
    public readonly verticalLines = input(1);
    public readonly verticalLinesHandler = input<TuiLineHandler>(TUI_ALWAYS_DASHED);
```

Refatoração - LC

- **Depois**:

- Inputs relacionados foram agrupados em objetos de configuração

- Lógica passou a consumir os objetos de configuração, tornando os conceitos do domínio explícito.

```
import {
    type TuiAxisXConfig,
    type TuiAxisYConfig,
    type TuiGridConfig,
} from './axes-config.interface';

export const TUI_ALWAYS_DASHED: TuiLineHandler = (index) =>
    (index && 'dashed') || 'solid';
export const TUI_ALWAYS_DOTTED: TuiLineHandler = (index) =>
    (index && 'dotted') || 'solid';
export const TUI_ALWAYS_SOLID: TuiLineHandler = () => 'solid';
export const TUI_ALWAYS_NONE: TuiLineHandler = () => 'none';

@Component({
    selector: 'tui-axes',
    templateUrl: './axes.template.html',
    styleUrl: './axes.style.less',
    changeDetection: ChangeDetectionStrategy.OnPush,
    host: {
        dir: 'ltr',
        '[class._centered]': 'centeredXLabels()',
    },
})
export class TuiAxes {
    // INPUTS - Agrupados por responsabilidade para melhor organização

    public readonly axisXLabels = input<ReadonlyArray<string | null>>([]);
    public readonly centeredXLabels = input(false);

    public readonly axisYLabels = input<readonly string[]>([]);
    public readonly axisYName = input('');
    public readonly axisYInset = input(false);
```

Refatoração - IIC

- **Problema:**
  - Nesse smell, classes herdam implementações completas de superclasses apenas para obter uma pequena parte do comportamento ou cumprir um contrato, criando hierarquias rígidas e fortemente acopladas.
- **Solução:**
  - A solução para o ICC é substituir herança por composição

Refatoração - IIC

- **Antes:**

```
4
5   import {AbstractTuiTableFilter} from './abstract-table-filter';
6
7   @Directive({
8       selector: '[tuiGenericFilter]',
9       providers: [tuiProvide(AbstractTuiTableFilter, TuiGenericFilter)],
10  })
11  export class TuiGenericFilter<T, G> extends AbstractTuiTableFilter<T, G> {
12      public readonly filter = input<(item: T, value: G) => boolean>(TUI_TRUE_HANDLER, {
13          alias: 'tuiGenericFilter',
14      });
15  }
```

Refatoração - IIC

- **Depois**:

```
4
5   import {AbstractTuiTableFilter} from './abstract-table-filter';
6
7   @Directive({
8       selector: '[tuiGenericFilter]',
9       providers: [tuiProvide(AbstractTuiTableFilter, TuiGenericFilter)],
10  })
11  export class TuiGenericFilter<T, G> implements AbstractTuiTableFilter<T, G> {
12      public readonly filter: Signal<(item: T, value: G) => boolean> = input<(item: T, value: G) => boolean>(TUI_TRUE_HANDLER, {
13          alias: 'tuiGenericFilter',
14      });
15  }
16      Ctrl+L to chat, Ctrl+K to generate
```

Refatoração - ANY

- **Problema:**
  - ocorre quando o tipo any é usado em excesso em um sistema TypeScript, fazendo com que o compilador perca sua capacidade de verificar tipos e detectar erros em tempo de compilação.
- **Solução:**
  - Uso de tipos genéricos para preservar flexibilidade sem perder segurança de tipo.
  - Substituição de any por unknown em casos onde o tipo exato não é conhecido, forçando validações explícitas.

Refatoração - ANY

- **Antes**:

```
export class TuiMobileDialogService extends TuiModalService<
    TuiMobileDialogOptions<any>,
    number
> {
    protected readonly options = inject(TUI_MOBILE_DIALOG_OPTIONS);
    protected readonly content = TuiMobileDialog;

    public override open(
        content: PolymorpheusContent<
            TuiPortalContext<TuiMobileDialogOptions<any>, number>
        >,
        options: Partial<TuiMobileDialogOptions<any>> = {},
    ): Observable<number> {
        return super.open(content, options);
    }
}
```

Refatoração - ANY

- **Depois**:

```
export class TuiMobileDialogService<I = unknown> extends TuiModalService<
    TuiMobileDialogOptions<I>,
    number
> {
    protected readonly options = inject(TUI_MOBILE_DIALOG_OPTIONS);
    protected readonly content = TuiMobileDialog;

    public override open(
        content: PolymorpheusContent<
            TuiPortalContext<TuiMobileDialogOptions<I>, number>
        >,
        options: Partial<TuiMobileDialogOptions<I>> = {},
    ): Observable<number> {
        return super.open(content, options);
    }
}
```

Resultados

- **Início:**
  - Code smells detectados: 365
  - Arquivos com code smells: 331
- **Final**:
  - Code smells detectados: 325
  - Arquivos com code smells: 314

Dificuldades

- Entender a estrutura do projeto.
- Pouca experiência com angular e refatoração.

Obrigado pela atenção!
Dúvidas?

UNIVERSIDADE FEDERAL DO CEARÁ
CAMPUS QUIXADÁ