**Exemplos de uso com explicação clara e precisa do vídeo "Curso Angular #14: Emitindo Eventos com Output properties"**

O vídeo "Curso Angular #14: Emitindo Eventos com Output properties" da Loiane Groner apresenta diversas técnicas para emitir eventos customizados em componentes Angular usando Output properties. Abaixo, detalhamos alguns dos principais exemplos de uso demonstrados no vídeo, com explicações claras e precisas:

**1. Criar um componente contador com botões para incrementar e decrementar um valor, emitindo eventos para atualizar a interface:**

**Cenário:** Imagine que você precisa de um componente reutilizável que permita aos usuários aumentar ou diminuir um valor numericamente, como a contagem de itens em um carrinho de compras. Além disso, você deseja que o componente emita eventos para que o componente pai possa atualizar a interface de acordo com a alteração do valor.

**Solução:** Para criar esse componente, siga estes passos:

1. **Crie um componente Angular:** Utilize o comando `ng generate component` para criar um novo componente, por exemplo, `contador-componente.component.ts`.
2. **Defina a estrutura HTML:** No arquivo `contador-componente.html`, defina a estrutura HTML do componente, incluindo os botões para incrementar e decrementar, e um campo de input para exibir o valor atual.

HTML

```
<div class="contador">
  <button class="btn btn-primary" (click)="incrementarValor()"> + </button>
  <input type="number" [(ngModel)]="valor">
  <button class="btn btn-primary" (click)="decrementarValor()"> - </button>
</div>
```

3. **Implemente a lógica no TypeScript:** No arquivo `contador-componente.component.ts`, implemente a lógica do componente, incluindo as propriedades, métodos, eventos personalizados e a emissão de eventos.

TypeScript

```typescript
import { Component, Input, Output, EventEmitter } from '@angular/core';

@Component({
  selector: 'app-contador-componente',
  templateUrl: './contador-componente.html',
  styleUrls: ['./contador-componente.css']
})
export class ContadorComponent {
  @Input() valorInicial: number = 0; // Valor inicial do contador
  valor: number = this.valorInicial; // Valor atual do contador

  @Output() mudouValor: EventEmitter<number> = new EventEmitter<number>(); //
Evento personalizado para emitir o valor alterado

  incrementarValor() {
    this.valor++;
    this.mudouValor.emit(this.valor);
  }

  decrementarValor() {
    this.valor--;
    this.mudouValor.emit(this.valor);
  }
}
```

4. **Utilize o componente em outro componente e escute o evento:** No componente pai, importe o `ContadorComponent` e utilize-o como um elemento HTML, passando os valores desejados para as propriedades `valorInicial` e escutando o evento `mudouValor` para atualizar a interface de acordo com a alteração do valor.

HTML

```html
<app-contador-componente [(valor)]="contador"
(mudouValor)="onValorMudou($event)"></app-contador-componente>
```

No método `onValorMudou` do componente pai, você pode atualizar a interface de acordo com o valor recebido no evento:

```
TypeScript
onValorMudou(valorAtual: number) {
  this.contador = valorAtual;
}
```

**2. Criar um componente de formulário com campos de entrada e botão de submit, emitindo eventos para validar e enviar os dados do formulário:**

**Cenário:** Suponha que você precise de um componente de formulário reutilizável com campos de entrada para nome, email e senha, e um botão de submit. O componente deve emitir eventos para validar os dados do formulário e enviar os dados para o componente pai quando o formulário for submetido.

**Solução:** Para criar esse componente, siga estes passos:

5. **Crie um componente Angular:** Utilize o comando `ng generate component` para criar um novo componente, por exemplo, `formulario-componente.component.ts`.
6. **Defina a estrutura HTML:** No arquivo `formulario-componente.html`, defina a estrutura HTML do formulário, incluindo os campos de entrada, rótulos e o botão de submit.

```
HTML
<form [formGroup]="formulario">
  <div class="form-group">
    <label for="nome">Nome:</label>
    <input type="text" formControlName="nome" class="form-control" id="
http://googleusercontent.com/youtube_content/21
```