Developing Custom Directives

With PAS 23.1.1, the new attribute **Custom Attributes** has been added to all supported form elements in the Designer to enable the usage of Angular directives. Angular attribute directives are used to change the appearance or behavior of DOM elements and Angular components. Refer to the official Angular documentation for details on how to create an Angular attribute directive.

Creating a Directive in Designer

Directives are created in your Designer development kit. The creation is similar to creating a form (refer to Developing Custom Forms in a Library > Creating a Form for details).

To create your own directive, switch to **<library-name>/projects/<library-name>/src/lib>** in your workspace project. You can now use the **@angular/cli** to create a new directive using the following command: `ng generate directive <directive-name>`

Step	Example
	The sample directive expands a text area when the user enters more then the allowed lines.

On this Page:

Form_Custom_Directive_Example

Click the icon to download a simple example model that shows how you can use **custom Angular directives** on form elements in **Scheer PAS** *Designer*.

Related Pages:

- Supported Form Elements

Related Documentation:

- Official Angular Documentation
  - Attribute Directives
  - Angular CLI

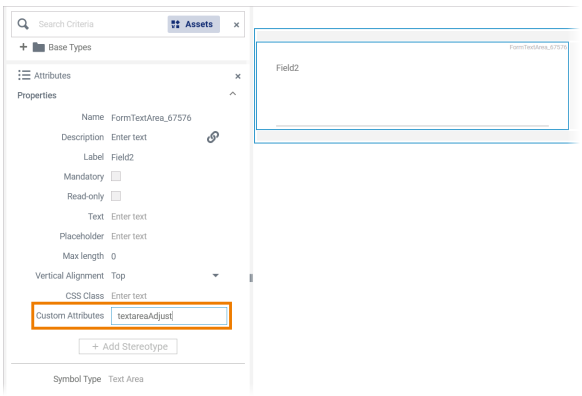
1	Write your directive in directory **lib** in your project and set it as input parameter in your code.
	
	Your custom attribute must be an input for this directive (see **textareaAdjust** in the example on the right).
		textarea-adjust.directive.ts
		
		`import {AfterViewInit, Directive, ElementRef, Input, OnDestroy, Renderer2} from '@angular/core';`
		
		`@Directive({`
		`  selector: '[textareaAdjust]'`
		`})`
		`export class TextareaAdjustDirective implements AfterViewInit, OnDestroy{`
		
		`  // Set your directive as input parameter in your code.`
		`  @Input() textareaAdjust: string = '';`
		`  listenerFn: () => void;`
		
		`  constructor(`
		`  private el: ElementRef,`
		`  private renderer: Renderer2`
		`  ) { }`
		
		`  ngAfterViewInit(): void {`
		`  const textarea = this.el.nativeElement.querySelector('textarea');`
		`  const handler = () => {`
		`  textarea.style.height = textarea.scrollHeight + 'px';`
		`  };`
		`  handler();`
		`  this.listenerFn = this.renderer.listen(textarea, 'blur', handler)`
		`  }`
		
		`  ngOnDestroy(): void {`
		`  if (this.listenerFn) {`
		`  this.listenerFn();`
		`  }`
		`  }`
		
		`}`

2	Write your directive into the imports and exports into the **form.module**.	**form.module.ts**
		`[...]`
		`@NgModule({`
		`  declarations: [FormComponent, TextareaAdjustDirective],`
		`    imports: [`
		`        FormRoutingModule,`
		`        AppCoreModule,`
		`        FormElementsModule,`
		`        MatTableModule,`
		`        TranslateModule,`
		`        ReactiveFormsModule`
		`    ],`
		`  exports: [FormComponent, TextareaAdjustDirective],`
		`  bootstrap: [FormComponent]`
		`})`
		`export class FormModule { }`

3	Write your directive into the **public-api**.	**public-api.ts**
		`[...]`
		`export * from './lib/textarea-adjust.directive';`

Now you are ready to use the directive in Designer form elements.

Using a Directive on a Form Element

	Open the form editor in the Designer. Click the form element to which you want to apply the directive and open its **Attibutes** panel. Enter the directive selector in the field **Custom Attributes**.

	ExpertAdvice You can define values for your directive like a **CustomAttribute=aValue**.
Field2 Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet. Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet.	During app execution, the directive is now applied to the form element.