FLOWUP

Structural directives

Vojtech Mašek

Head dev @FlowUp

/ vmasek

/ VojtechMasek

/ vmasek

Common directives

*ngIf

*ngFor

What do they do

- are responsible for HTML layout
- (re)shape the DOM's structure
  - typically by adding, removing, or manipulating elements

View Container reference

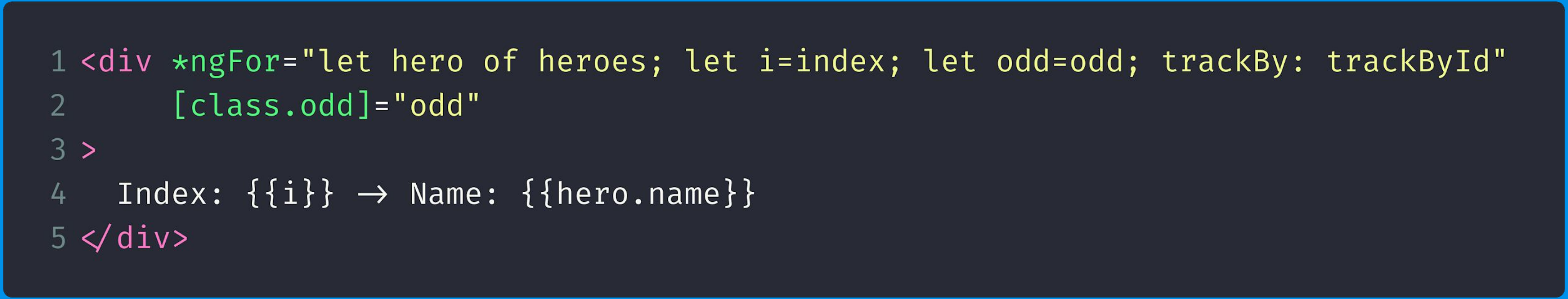
- represents a container
- one or more views can be attached

FLOWUP

Template reference

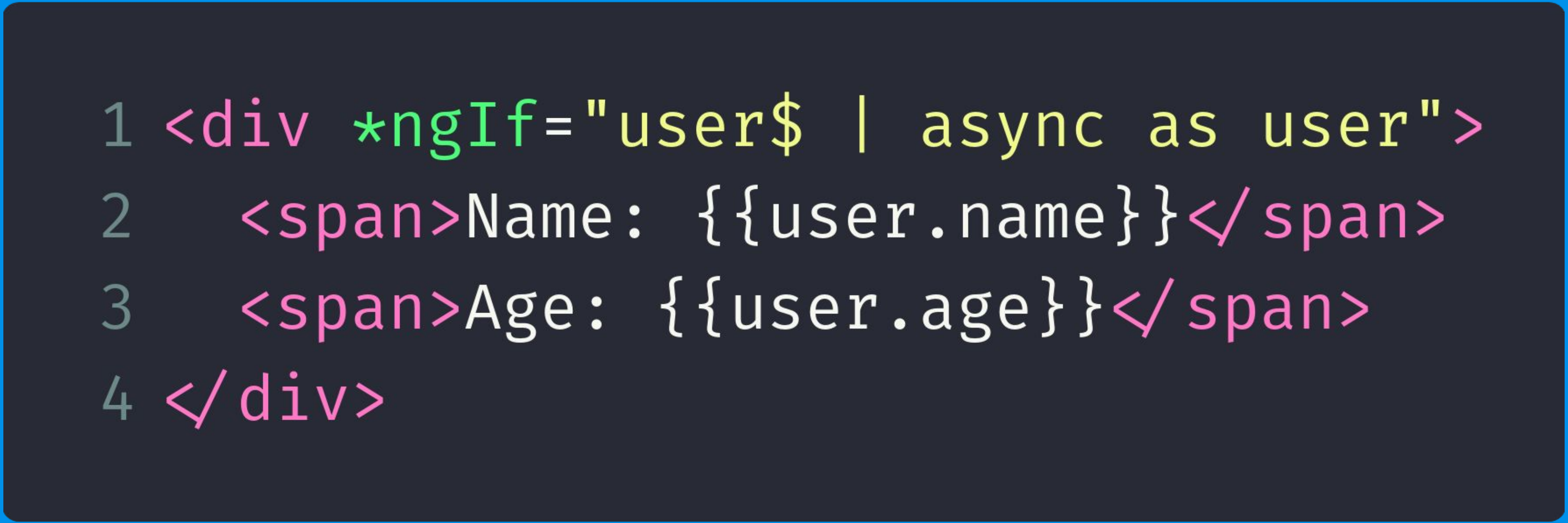
- represents an embedded template
  - can be used to instantiate embedded views
- element that we’ve put the directive on

Context variables



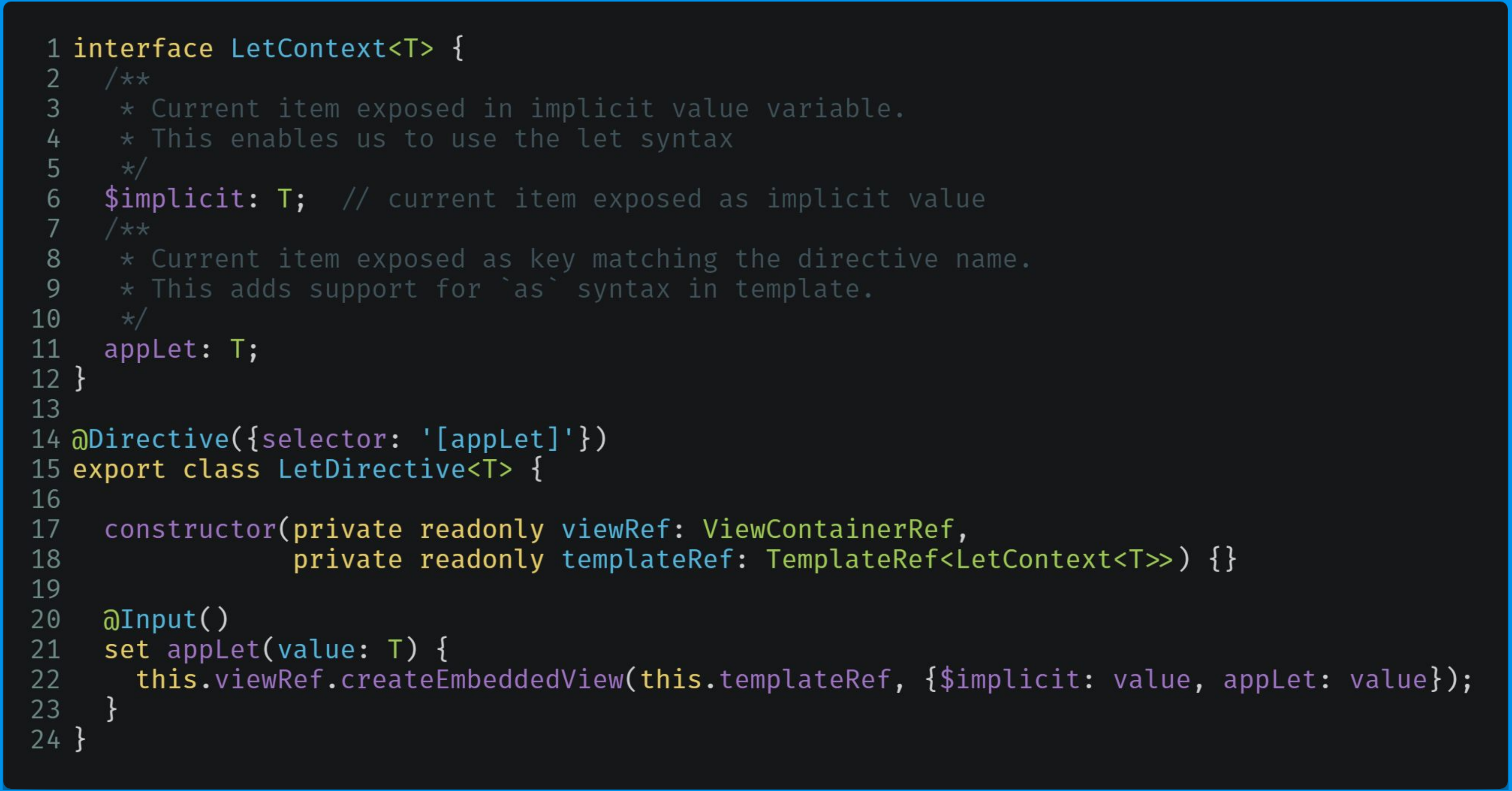
```
<div *ngFor="let hero of heroes; let i=index; let odd=odd; trackBy: trackById"
     [class.odd]="odd"
>
  Index: {{i}} -> Name: {{hero.name}}
</div>
```

In template variable declaration



```html
<div *ngIf="user$ | async as user">
  <span>Name: {{user.name}}</span>
  <span>Age: {{user.age}}</span>
</div>
```

Let directive



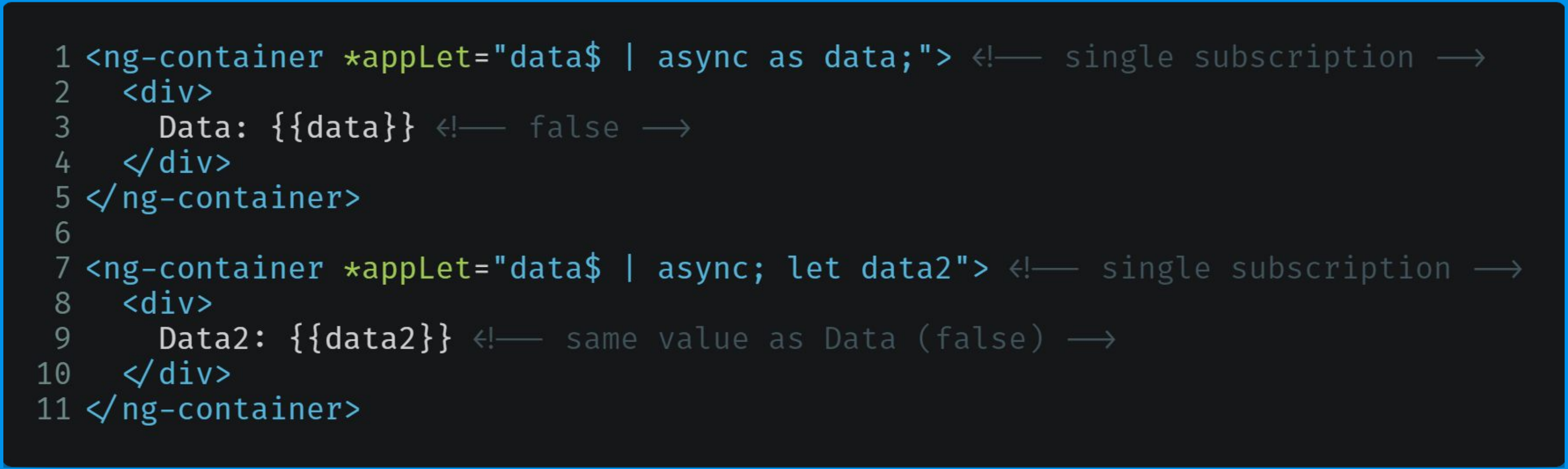
```
interface LetContext<T> {
  /**
   * Current item exposed in implicit value variable.
   * This enables us to use the let syntax
   */
  $implicit: T;  // current item exposed as implicit value
  /**
   * Current item exposed as key matching the directive name.
   * This adds support for `as` syntax in template.
   */
  appLet: T;
}

@Directive({selector: '[appLet]'})
export class LetDirective<T> {

  constructor(private readonly viewRef: ViewContainerRef,
              private readonly templateRef: TemplateRef<LetContext<T>>) {}

  @Input()
  set appLet(value: T) {
    this.viewRef.createEmbeddedView(this.templateRef, {$implicit: value, appLet: value});
  }
}
```

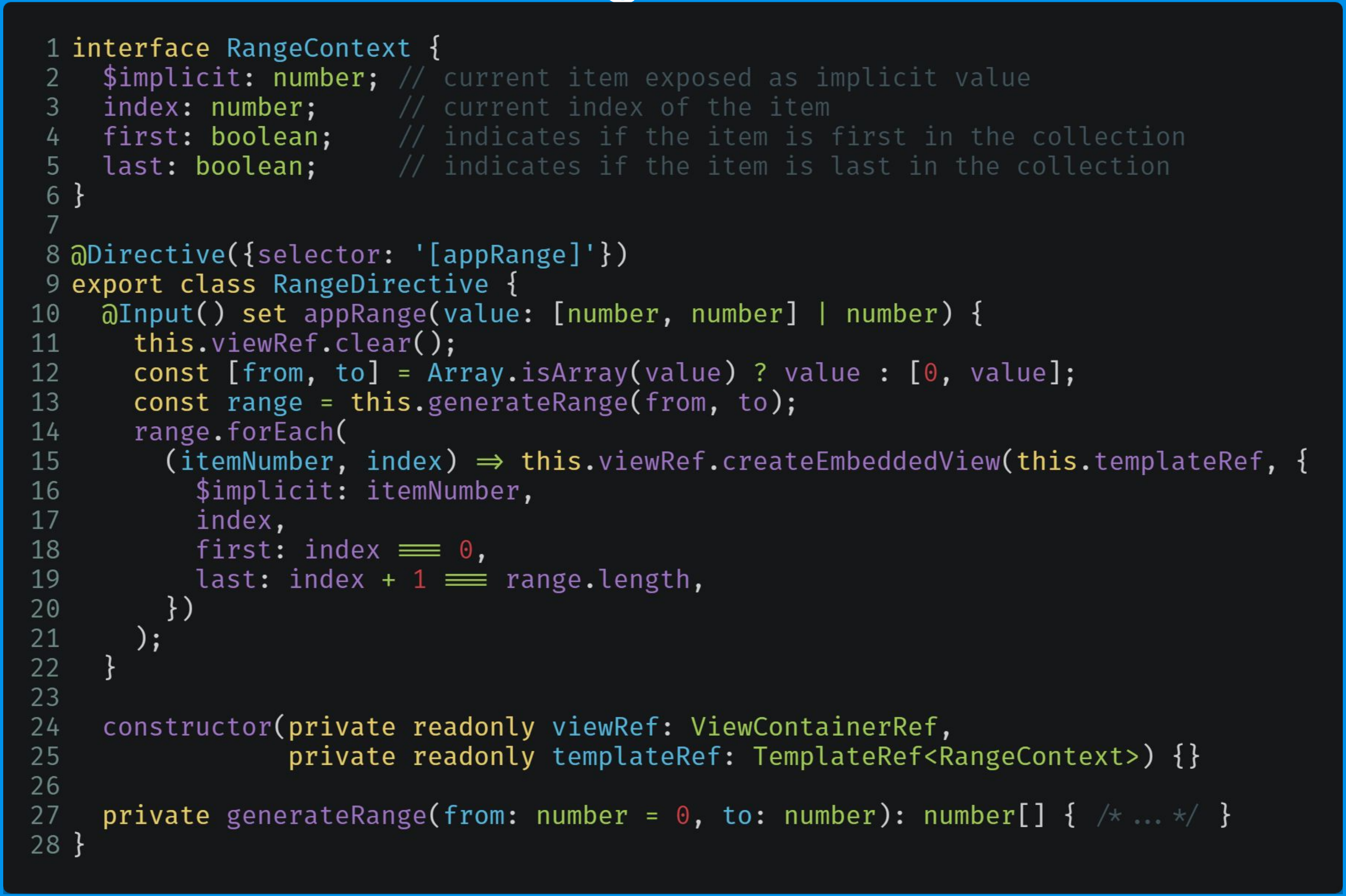
Let directive usage



```html
<ng-container *appLet="data$ | async as data;"> <!-- single subscription -->
  <div>
    Data: {{data}} <!-- false -->
  </div>
</ng-container>

<ng-container *appLet="data$ | async; let data2"> <!-- single subscription -->
  <div>
    Data2: {{data2}} <!-- same value as Data (false) -->
  </div>
</ng-container>
```

For range directive



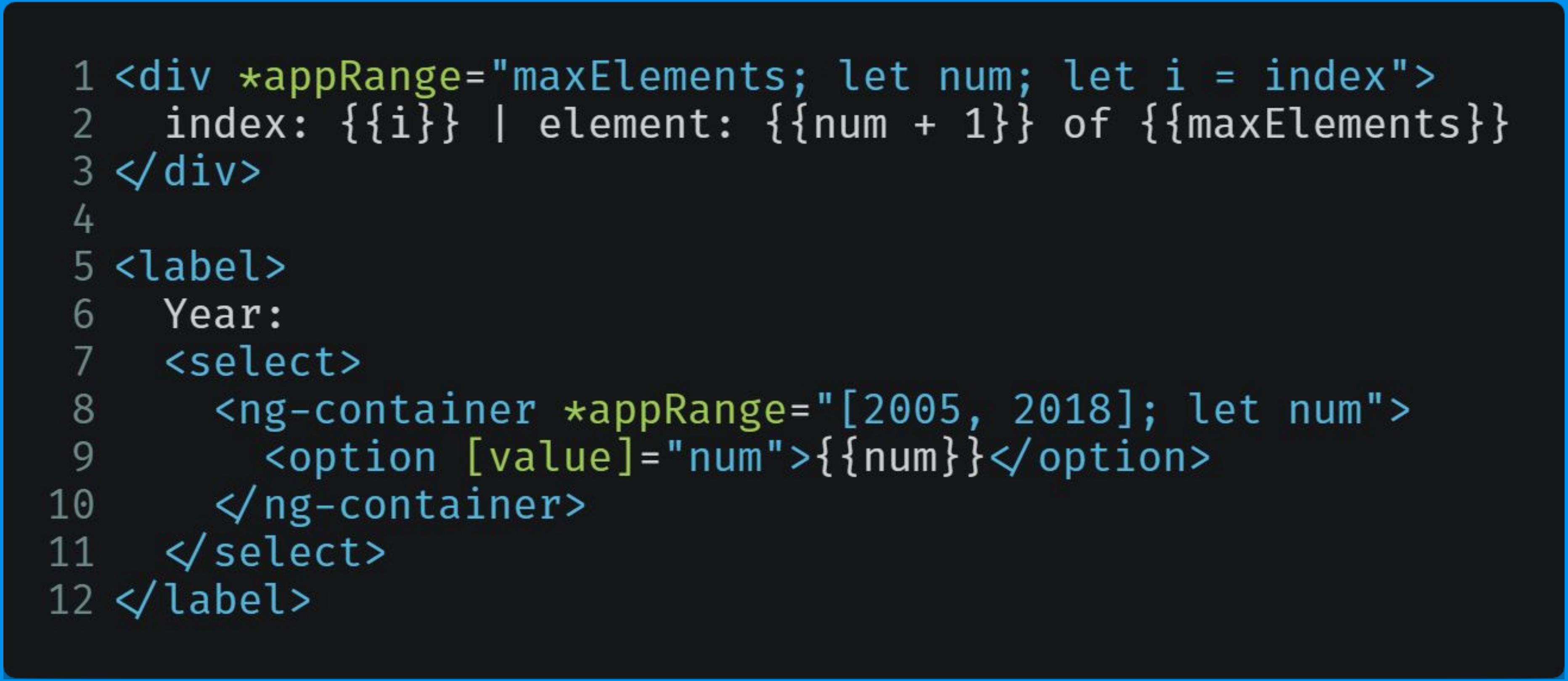
```typescript
interface RangeContext {
  $implicit: number; // current item exposed as implicit value
  index: number;     // current index of the item
  first: boolean;    // indicates if the item is first in the collection
  last: boolean;     // indicates if the item is last in the collection
}

@Directive({selector: '[appRange]'})
export class RangeDirective {
  @Input() set appRange(value: [number, number] | number) {
    this.viewRef.clear();
    const [from, to] = Array.isArray(value) ? value : [0, value];
    const range = this.generateRange(from, to);
    range.forEach(
      (itemNumber, index) => this.viewRef.createEmbeddedView(this.templateRef, {
        $implicit: itemNumber,
        index,
        first: index === 0,
        last: index + 1 === range.length,
      })
    );
  }

  constructor(private readonly viewRef: ViewContainerRef,
              private readonly templateRef: TemplateRef<RangeContext>) {}

  private generateRange(from: number = 0, to: number): number[] { /* ... */ }
}
```

For range directive usage



```html
<div *appRange="maxElements; let num; let i = index">
  index: {{i}} | element: {{num + 1}} of {{maxElements}}
</div>

<label>
  Year:
  <select>
    <ng-container *appRange="[2005, 2018]; let num">
      <option [value]="num">{{num}}</option>
    </ng-container>
  </select>
</label>
```

FLOWUP

tiny.cc/structural-directives

Vojtech Mašek

Head dev @FlowUp

/ vmasek

/ VojtechMasek

/ vmasek