ANGULAR

PIPES

LUXOFT

LXFT LISTED NYSE

USING PIPES

```
import {Component} from 'angular2/core'

@Component({
   selector: 'hero-birthday',
   template: `<p>The hero's birthday is
         {{ birthday | date: ‘dd.MM.yyyy’ }}</p>`
})
export class HeroBirthday {
   birthday = new Date(1988,3,15); // April 15, 1988
}
```

DEFINE CUSTOM PIPE

```
import { Pipe, PipeTransform } from '@angular/core';
    /* Raise the value exponentially
     * Takes an exponent argument that defaults to 1.
     * Usage:
     *   value | exponentialStrength:exponent
     * Example:  {{ 2 |  exponentialStrength:10}} formats to: 1024 */
    @Pipe({name: 'exponentialStrength'})
    export class ExponentialStrengthPipe
       implements PipeTransform {

      transform(value: number, exponent: string): number {
        let exp = parseFloat(exponent);
        return Math.pow(value, isNaN(exp) ? 1 : exp);
      }
    }
```

USE CUSTOM PIPE

```
import {Component} from '@angular/core';

@Component({
   selector: 'power-booster',
   template: `
   <h2>Power Booster</h2>
   <p>
    Super power boost: {{2 | exponentialStrength: 10}}
   </p>`
})
export class PowerBooster { }
```

USE ASYNC PIPE

```
import {Component} from '@angular/core';

// Initial view: "Message: "
// After 500ms: Message: You are my Hero!"

@Component({
  selector: 'hero-message',
  template: `Message: {{delayedMessage | async}} {{myObj|json}} `,
})
export class HeroAsyncMessageComponent {
  myObj = {a:10,b:20}
  arr = [1,2,3,4,5]
  delayedMessage:Promise<string> =
      new Promise((resolve, reject) => {
        setTimeout(() => resolve('You are my Hero!'), 500);
      });
}
```

PURE AND IMPURE PIPES

There are two categories of pipes: ***pure*** and ***impure***. Pipes are **pure by default**.

Angular executes a ***pure pipe*** only when it detects a **pure change** to the input value. A pure change is either a change to a ***primitive input value*** (String, Number, Boolean, Symbol) or a changed ***object reference*** (Date, Array, Function, Object). Angular ignores changes within (composite) objects.

Angular executes an ***impure pipe*** during ***every component change detection cycle***. An impure pipe is called often, as often as every keystroke or mouse-move. With that concern in mind, implement an impure pipe with great care. An expensive, long-running pipe could destroy the user experience.

Example: <div *ngFor="let note of notes">**{note|with-date}**

…somewhere in code: note.lastUpdated = new Date();

```
@Pipe({name: 'with-date', pure: false})
export class NoteWithDatePipe
        implements PipeTransform {
        transform(note: Note): string {
                return note.text+' ('+ note.lastUpdated+')';
        }
}
```