

Overview

currency	Supported
date	Supported
uppercase	Supported
json	Supported
limitTo	Supported
lowercase	Supported
number	Not
orderBy	Not
filter	Not
async	Supported
decimal	Supported
percent	Supported

A Pipe is a filter applied over a angular expression. This is denoted by the | (pipe symbol)

Example Code

```
<!-- Sep 1, 2015 -->
<p>{{date | date:'mediumDate'}}</p>
<!-- September 1, 2015 -->
<p>{{date | date:'yMMMMd'}}</p>
<!-- 3:50 pm -->
<p>{{date | date:'shortTime'}}</p>
```

Currency

Usage	expression currency[:currency-Code[:symbolDisplay[:digitInfo]]]
current-cyCode	ISO 4217 Compliant, eg USD, EUR

```
class CurrencyPipe {
  transform(value: any, currencyCode?:
  string, symbolDisplay?: boolean, digits?:
  string) : string
}
```

Date

Usage	expression date[:format]
Year	y or yy
Month	M or MM
Day	d or D
Weekday	EEE or EEEE
Hour	j or jj
Hour 12	h or hh
Hour 24	H or HH
Minute	m or mm
Second	s or ss
Timezone	z

```
class DatePipe {
  transform(value: any, pattern?: string) :
  string
  supports(obj: any) : boolean
}
```

Uppercase

Usage	item uppercase
-------	------------------

Json

Usage	item json
-------	-------------

Transforms any input value using JSON.stringify. Useful for debugging.

lowercase

Usage	item lowercase
-------	------------------

async

Usage	item async
-------	--------------

The async pipe subscribes to an Observable or Promise and returns the latest value it has emitted. When a new value is emitted, the async pipe marks the component to be checked for changes.

decimal

Usage	expression number[:digitInfo]
digit Info	{minIntegerDigits}.{minFractionDigits}-{maxFractionDigits}

minIntegerDigits is the minimum number of integer digits to use. Defaults to 1.
minFractionDigits is the minimum number of digits after fraction. Defaults to 0.
maxFractionDigits is the maximum number of digits after fraction. Defaults to 3.

percent

Usage	expression percent[:digitInfo]
	see decimal for usage

Custom Pipes

```
import {Component, View,
bootstrap, Pipe, PipeTransform}
from 'angular2/angular2';
@Pipe({
  name: 'tempC onvert'
})
class TempCo nvertPipe
implements PipeTr ansform {
  tra nsf orm (value: number,
args: any[]) {
    if( value && !isNaN -
(value) && args[0] ===
'celsius') {
      var temp = (value -
32) * 5/9;
      var places = args[1];
      return temp.t oFi -
xed (pl aces) + ' C';
    }
    return;
  }
}
```



By **Nathan** (Nathane2005)

cheatography.com/nathane2005/

Published 23rd May, 2016.

Last updated 22nd May, 2016.

Page 1 of 2.

Sponsored by **Readable.com**

Measure your website readability!

<https://readable.com>

Using non standard pipes

import	import as normal
component	pipes: [ExponentialStrengthPipe]
usage	



By **Nathan** (Nathane2005)

cheatography.com/nathane2005/

Published 23rd May, 2016.

Last updated 22nd May, 2016.

Page 2 of 2.

Sponsored by **Readable.com**

Measure your website readability!

<https://readable.com>