Expressions and Interpolation

Expression: {{ exp }}

Something that evaluates to some value

- ✧ Processed by Angular & *roughly* similar to the result *of eval(some_js)*
- ✧ Executed in the context of the scope & has access to properties on $scope
- ✧ Doesn’t throw errors if it results in a TypeError or ReferenceError
- ✧ Control flow functions (e.g., ‘if’ statements, etc.) are not allowed
- ✧ Accept a filter or a filter chain to format the output

Interpolation

Process of evaluating a *string literal* containing one or more placeholders, which are replaced with values

- In Angular, this string:

  ```
  Message is {{ message }}
  ```

  (provided message = “hello”) is interpolated into this string:

  ```
  Message is hello
  ```

Interpolation

Process of evaluating a *string literal* containing one or more placeholders, which are replaced with values

- ✧ Still connected to the original *message* property
  - If $scope.message changes, so will the interpolation result

Summary

- ✧ Expression: something that turns into a value
  - {{ expression }}
  - Angular expressions are tied to the scope they are in
  - Doesn’t display TypeError and ReferenceError to user
- ✧ Interpolation: replaces placeholders in a string with values
  - In Angular, these placeholders are usually expressions
  - Result is automatically updated when placeholder value changes