HTML is a set of written instructions for *how* to display a web page.

The browser reads the HTML and creates something called a *DOM*, a *Document Object Model*. This is the manifestation of those HTML instructions in memory.

Changing the HTML doesn't automatically update the webpage unless the user refreshes the browser, changing the DOM however instantly updates the webpage.

There is mostly a 1-to-1 mapping between the names and values of HTML attributes and their equivalent DOM properties, but not always.

The hidden HTML attribute is a good example, it only needs to `exist` on an HTML element to instruct the browser to hide the element.

So `hidden="true"` hides the element but confusingly so does `hidden="false"` in HTML we just need to add `hidden` to hide the element.

The DOM representation of the `hidden` attribute is a property also called `hidden`, which if set to `true` hides the element and `false` shows the element.

Angular doesn't manipulate HTML attributes, it manipulates DOM properties because the *DOM* is what actually gets displayed.

So when we write `[hidden]` we are manipulating the *DOM property* and not the *HTML attribute*.

This is why the above is called Input *Property* Binding and not Input Attribute Binding.

Input Property Binding

Looking back at our use of the hidden property:

```
<p class="card-text" [hidden]="true">{{joke.punchline}}</p>
```

The *target* inside `[]` is the name of the property. In the example above the target is the `hidden` DOM property.

The *text* to the right of `=` is javascript code that gets executed and the resulting value is assigned to the target.

`true` is still javascript code which if executed returns `true`.

So in summary, we are binding to the DOM property `hidden` and setting it to `true` so the element is hidden.

In other parts of the web you'll see this referred to as just *property binding*. However to distinguish it from the other type of binding in Angular I like to call this **input** property binding.

We can only use this type of binding to change the value of the target. We can't use it to get *notified*