Me:

in my code there is a part of code that repeats itself multiple times how can i group the common repeation but still get the same result in html and css. The mentioned part of code is:

```css
.language-card:hover#powershell ~ .programming-language-difficulty-rating .difficulty-ra
.language-card:hover#powershell ~ .programming-language-difficulty-rating .difficulty-ra
.language-card:hover#powershell ~ .programming-language-difficulty-rating .difficulty-ra
    background-image: url("../../images/full_star.png");
}
.language-card:hover#powershell ~ .programming-language-difficulty-rating .difficulty-ra
    background-image: url("../../images/half_star.png");
}

/* Experience Rating */
.language-card:hover#powershell ~ .programming-language-experience .experience-star:nth-
.language-card:hover#powershell ~ .programming-language-experience .experience-star:nth-
.language-card:hover#powershell ~ .programming-language-experience .experience-star:nth-
.language-card:hover#powershell ~ .programming-language-experience .experience-star:nth-
    background-image: url("../../images/full_star.png");
}
```

ChatGPT:

If you have 11 blocks of similar code like the one you provided, you can use CSS variables or create reusable classes to reduce the repetition and manage it more efficiently. However, because you're targeting specific elements with IDs like `#powershell` or `#c`, a direct reuse is limited without JavaScript. But, you can still optimize it to minimize repetition using `data-attributes` or mixins if you're using a CSS preprocessor like SASS/LESS.

Here’s how you can refactor it for a cleaner approach, assuming you want to stick with plain CSS:

Approach 1: Using Common Class for Shared Styles

Instead of repeating blocks for each language like `#powershell` or `#c`, you can create a reusable class and use `data-attributes` or a shared `class` on the language cards to avoid hard-coding IDs repeatedly.

```html
<!-- HTML Example -->
<div class="language-card" data-language="powershell">
  <!-- Content -->
</div>

<div class="language-card" data-language="c">
  <!-- Content -->
</div>
```

Optimized CSS with Data Attributes:

You can target the elements dynamically using the **`data-language`** attribute, which allows for more flexibility.

```css
/* Common Full Star for Difficulty and Experience */
.language-card:hover[data-language="powershell"] ~ .programming-language-difficulty-rati
.language-card:hover[data-language="c"] ~ .programming-language-difficulty-rating .diffi
.language-card:hover[data-language="powershell"] ~ .programming-language-experience .expe
.language-card:hover[data-language="c"] ~ .programming-language-experience .experience-s
    background-image: url("../../images/full_star.png");
}

/* Common Half Star for Difficulty */
.language-card:hover[data-language="powershell"] ~ .programming-language-difficulty-rati
.language-card:hover[data-language="c"] ~ .programming-language-difficulty-rating .diffi
    background-image: url("../../images/half_star.png");
}
```

Explanation:

1. **`data-language` attribute**: This allows you to target different programming languages dynamically without repeating the same styles for each specific ID. The value of **`data-language`** (like "powershell" or "c") can be customized for each language.
2. **Shared rules**: Both the full star and half star rules for **`difficulty`** and **`experience`** ratings are now applied dynamically based on the **`data-language`** value, avoiding repetition of individual IDs.

Approach 2: Using SASS/LESS (If Available)

If you're using a CSS preprocessor, you can make this even more modular by using mixins or loops. For example, using SASS:

```
@each $language in powershell, c, java, python {
  /* Full Star */
  .language-card:hover##{$language} ~ .programming-language-difficulty-rating .difficulty
  .language-card:hover##{$language} ~ .programming-language-experience .experience-star:n
    background-image: url("../../images/full_star.png");
  }

  /* Half Star */
  .language-card:hover##{$language} ~ .programming-language-difficulty-rating .difficulty
    background-image: url("../../images/half_star.png");
  }
}
```

When you have 11 or more blocks:

- **With `` `data-language` `` approach**, you'd only need to add one more **`` `data-language` ``** value per new language. The CSS doesn't grow significantly.
- **If using a preprocessor**, loops would make this even more scalable.

This should make your CSS more maintainable and easier to extend for multiple languages.

---

*Powered by **https://markdowntoimage.com***