Front End Tricks

part 1

1. Document flow
2. Inline-block gap
3. Specificity

Document flow

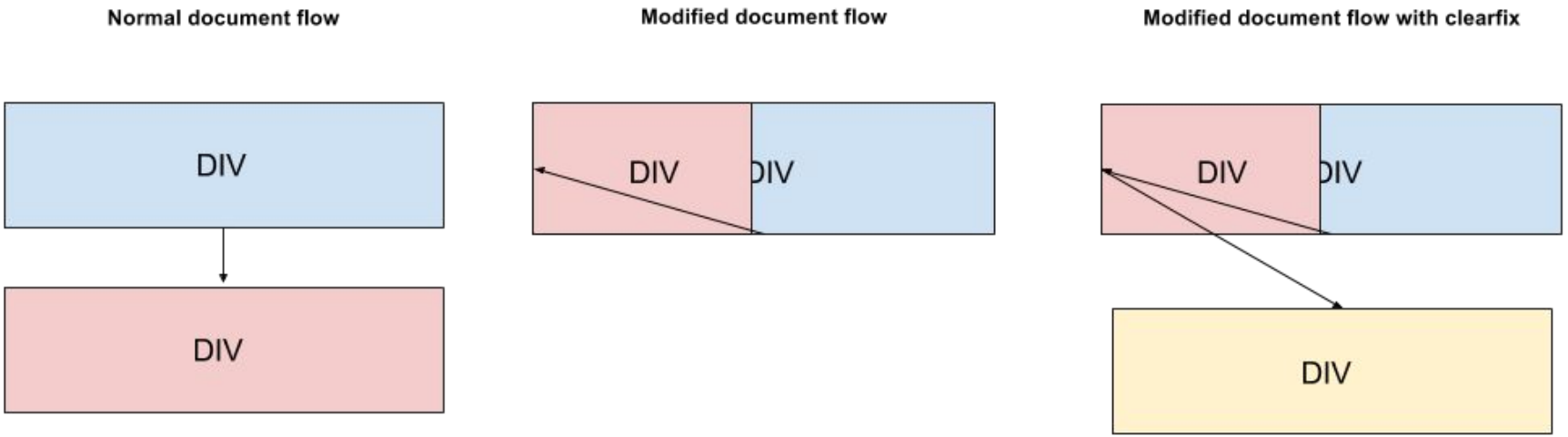
Normal document flow

Definition by W3C:

“Boxes in the normal flow belong to a formatting context, which may be block or inline, but not both simultaneously. Block-level boxes participate in a block formattingcontext. Inline-level boxes participate in an inline formatting context.”

http://www.w3.org/TR/CSS21/visuren.html#normal-flow

Document flow - visualization



Clear-fix technique

Classic resetting document flow

```
<div>
    <div style="float: left;">Sample DIV</div>
    <div style="clear: both;"></div>
</div>
```

Clear-fix technique

```
.clearfix:after {
   content: " ";
   visibility: hidden;
   display: block;
   height: 0;
   clear: both;
}

<div class="clearfix">
   <div style="float: left;" class="clearfix">Sample
DIV</div>
</div>
```

Perks: no additional tag at the end

Conclusion

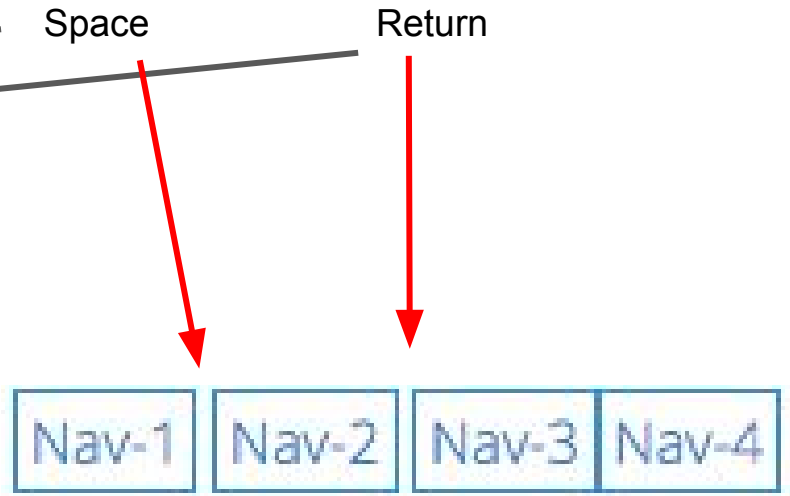
Keep in mind what is the normal flow of the document and how it can be impacted. If you'll face already modified document flow, you now know how to restore it.

Inline-block gap

Inline-block gap - causes

```
<nav>
  <a href="#">Nav-1</a> <a href="#">Nav-2</a>
  <a href="#">Nav-3</a><a href="#">Nav-4</a>
</nav>

nav a {
  display: inline-block;
  padding: 2px 3px;
}
```



How to fix it ?

Skip the white spaces between the markups (between the closing and opening braces should be no space in any form)

```
<nav>
  <a href="#">Nav-1</a><a href="#">Nav-2</a><a href="#">Nav-3</a><a href="#">Nav-4</a>
</nav>
```

Set negative margins between the particular elements.

```
nav a {
  display: inline-block;
  padding: 2px 3px;
  margin-left: -4px;
}
```

How to fix it ?

Remove the closing tags

```
<nav>
  <a href="#">Nav-1
  <a href="#">Nav-2
  <a href="#">Nav-3
  <a href="#">Nav-4
</nav>
```

Set to the container property font-size: 0

```
nav a {
  display: inline-block;
  padding: 2px 3px;
  font-size: 12px;
}

nav {
  font-size: 0;
}
```

Conclusion

This is the most common puzzler for any developers who start to code any basic front-end markups and you can waste a lot of time wondering about this empty gap. Remember that this is a native behaviour of the CSS styles and know how you can fix it.

Specificity

Specificity values

Selector values

- inline styles - x10000
- ids - x1000
- classes, pseudo classes, attributes - x100
- elements, pseudo-elements - x1

order of the styles makes no difference

Specificity examples

```
// specificity = 1 + 1 = 2

nav a {}

// specificity = 1000 + 100 = 1100
#my-account .login

// specificity = 1 + 1 = 2
a::before

// specificity = 100 + 100 + 100 = 300
.nav-button.nav-button.nav-button
```

```
nav.my-nav a {
    color: #000099; // 0012
  }

  nav.my-nav a.nav-button {
    color: #009999; // 0022
  }

  .nav-button.nav-button.nav-button {
    color: #990000; // 0030 - the strongest
  }

<nav class="my-nav">
  <a class="nav-button" href="#">Nav-1</a>
  <a class="nav-button" href="#">Nav-2</a>
  <a class="nav-button" href="#">Nav-3</a>
</nav>
```

Exceptions to the rule

The CSS styles are applied along with the specificity values and there are no exceptions from it, except…

- inline styles
- !important flag (the strongest of all) - the last style with important flag is applied (order makes the difference)

Conclusion

The specificity is one of the most fundamental concepts in regards to the CSS. The knowledge of how styles are being applied can save a lot of time.

Thank you :)