Conflicts – *For a given property, two or more CSS rules select an element and set different values.*

* Often the same element may be selected by more than one rule.
* The problem is often not so obvious.

Causes of Conflict

* Carelessness
* When you use multiple stylesheets written by different people
* But often they are not mistakes – they are deliberate:
* Web developers exploit conflicts because they can lead to more concise or more easily understood stylesheets

Resolution of Conflicts

Resolution uses 3 criteria, taken in this order:

1. Origin
2. Specificity
3. Ordering

But note that this is a simplified explanation

Origin

CSS rules can have different origins:

* Author stylesheets – the ones we write
* User-agent stylesheets – the browser’s default stylesheet

It is because of the user-agent stylesheet that by default, e.g.

* em & I elements are in italic
* strong, b & h1-h6 elements are in bold
* h1 headings are bigger than h2 headings
* consecutive paragraphs are separated by a blank line

If two rules conflict and one is from and author stylesheet and the other is from the user-agent stylesheet, then the author stylesheet wins.

Specificity

If origin does not resolve a conflict, then we turn to specificity.

The more specific rule wins.

Roughly,

* Selecting by id is the most specific
* Selecting by class is the second most specific
* Then, a selector that is longer is more specific.
* \* and combinations have no effect on specificity

Ordering  
If origin and specificity do not resolve a conflict, then we turn to ordering.

The later rule wins.

Ordering (Usually Comes Up As a Question)

Consider the three ways of applying styles to a web page:

* Inline styles are thought as coming later than embedded stylesheets, which come later than external stylesheets
* Suppose there is more than one external stylesheet
* Then the order of the link elements is used

The Cascade

* Note the way that conflict resolution cascades:

1. From origin
2. … to specificity
3. … to ordering

* But keep in mind this is only for cases of conflict
* A classic mistake is to use the cascade when you should instead use inheritance and the blocking of inheritance`

Exploiting Cascades

* CSS experts exploit blocking of inheritance and conflicts to get a more elegant stylesheet
* E.g. suppose we want all i elements to be blue, except ones referring to a video games, which should be red

Tips

* You need to learn this stuff for the exam
* If you hover a selector in VSCode, it shows you the specificity
* Chrome Developer Tools shows CSS that gets ‘eliminated’.