Chapter 47: Feature Queries

Parameter	Details
`(property: value)`	Evaluates true if the browser can handle the CSS rule. The parenthesis around the rule are required.
`and`	Returns true only if both the previous and next conditions are true.
`not`	Negates the next condition
`or`	Returns true if either the previous or next condition is true.
`(...)`	Groups conditions

Section 47.1: Basic @supports usage

```
@supports (display: flex) {
  /* Flexbox is available, so use it */
  .my-container {
    display: flex;
  }
}
```

In terms of syntax, `@supports` is very similar to `@media`, but instead of detecting screen size and orientation, `@supports` will detect whether the browser can handle a given CSS rule.

Rather than doing something like `@supports (flex)`, notice that the rule is `@supports (display: flex)`.

Section 47.2: Chaining feature detections

To detect multiple features at once, use the `and` operator.

```
@supports (transform: translateZ(1px)) and (transform-style: preserve-3d) and (perspective: 1px) {
  /* Probably do some fancy 3d stuff here */
}
```

There is also an `or` operator and a `not` operator:

```
@supports (display: flex) or (display: table-cell) {
  /* Will be used if the browser supports flexbox or display: table-cell */
}
@supports not (-webkit-transform: translate(0, 0, 0)) {
  /* Will *not* be used if the browser supports -webkit-transform: translate(...) */
}
```

For the ultimate `@supports` experience, try grouping logical expressions with parenthesis:

```
@supports ((display: block) and (zoom: 1)) or ((display: flex) and (not (display: table-cell))) or
(transform: translateX(1px)) {
  /* ... */
}
```

This will work if the browser

1. Supports `display: block` AND `zoom: 1`, or
2. Supports `display: flex` AND NOT `display: table-cell`, or
3. Supports `transform: translateX(1px)`.