@supports

1. What is @supports?

2. What is the syntax?

3. When and why was it created?

4. Which browsers use it?

5. What are vendor prefixes?

6. Does @supports replace them?

Content Management Seminar

by Nicholas de Havilland

@supports

1. What is @supports?

2. What is the syntax?

3. When and why was it created?

4. Which browsers use it?

5. What are vendor prefixes?

6. Does @supports replace them?

What is @supports?

- It is called a **feature query**, which include the familiar **@media** rule used in graceful degradation or progressive enhancement

- It helps the browser to test CSS rules, and determine if it can understand them without having to parse and apply the entire declaration block

- Similar to a media query, it allows the developer to create declarations that are executed only if supported by the browser, and ignored otherwise

- It is part of the CSS3 Conditional Rules Specification:

  https://drafts.csswg.org/css-conditional-3

@supports

1. What is @supports?

2. What is the syntax?

3. When and why was it created?

4. Which browsers use it?

5. What are vendor prefixes?

6. Does @supports replace them?

What is the syntax?

- The rule may be placed at the top level of your code or nested inside any other conditional **at-rule**

- It tests the condition specified in brackets and continues to parse if it returns a Boolean value of **true**, or ignores the entire block if **false**

- Multiple conditions can be combined using logical operators: AND, OR and NOT

- As a function to test whether combinator/pseudo selectors are supported (>, :first-child,…) or even custom properties

@supports

1. What is @supports?
2. What is the syntax?
3. When and why was it created?
4. Which browsers use it?
5. What are vendor prefixes?
6. Does @supports replace them?

What is the syntax?

```
/* example 1: first level */
@supports (property: value) {
   selector { property: value; }
   …
}

/* example 2: first level*/
@media (orientation: landscape) {
   selector { property: value; }
   …
   /* example 2: second level*/
   @supports (property: value) {
      selector { property: value; }
      …
   }
}
```

@supports

1. What is @supports?

2. What is the syntax?

3. When and why was it created?

4. Which browsers use it?

5. What are vendor prefixes?

6. Does @supports replace them?

What is the syntax?

```
/* example 3: not */
@supports not (property: value) {
    selector { property: value; }
    …
}
/* example 4: and */
@supports (property: value) and (property: value) {
    selector { property: value; }
    …
}
/* example 5: or */
@supports (property: value) or (property: value) {
    selector { property: value; }
    …
}
/* example 6: multiple */
@supports ((property: value) and (property: value))
                               or (property: value) {
    selector { property: value; }
    …
}
```

@supports

1. What is @supports?

2. What is the syntax?

3. When and why was it created?

4. Which browsers use it?

5. What are vendor prefixes?

6. Does @supports replace them?

What is the syntax?

```
/* example 7: direct child combinator */
@supports selector(A > B) {
    selector { property: value; }
    …
}

/* example 8: pseudo element selector */
@supports selector(::-webkit-scrollbar-thumb) {
    selector { property: value; }
    …
}
```

@supports

1. What is @supports?

2. What is the syntax?

3. When and why was it created?

4. Which browsers use it?

5. What are vendor prefixes?

6. Does @supports replace them?

When and why was it created?

- Feature detection was originally carried our client-side using JavaScript, not implemented in CSS
- It was not available as a CSS rule in major browsers until 2013, with versions of Firefox from 22, and Chrome from 28.
- Implementation in CSS increases website performance by eliminating the additional JavaScript polyfills
- Allows layout enhancement by placing less reliably supported features after the more widely supported fallbacks
- This provides a robust and modular way of catering for accessibility in differing platforms and form factors

@supports

1. What is @supports?

2. What is the syntax?

3. When and why was it created?

4. Which browsers use it?

5. What are vendor prefixes?

6. Does @supports replace them?

When and why was it created?

```
/* fallback 1: float rules */
selector { property: value; }
…

/* fallback 2: flex rules */
@supports (display: flex) {
   selector { property: value; }
   …
}

/* fallback 3: grid rules */
@supports (display: grid) {
   selector { property: value; }
   …
}
```

@supports

1. What is @supports?

2. What is the syntax?

3. When and why was it created?

4. Which browsers use it?

5. What are vendor prefixes?

6. Does @supports replace them?

When and why was it created?

```
/* flex layout only if there is no grid support */
@supports (display: flex) and not (display: grid) {
   selector { property: value; }
   …
}

/* grid layout */
@supports (display: grid) {
   selector { property: value; }
   …
}
```

@supports

1. What is @supports?

2. What is the syntax?

3. When and why was it created?

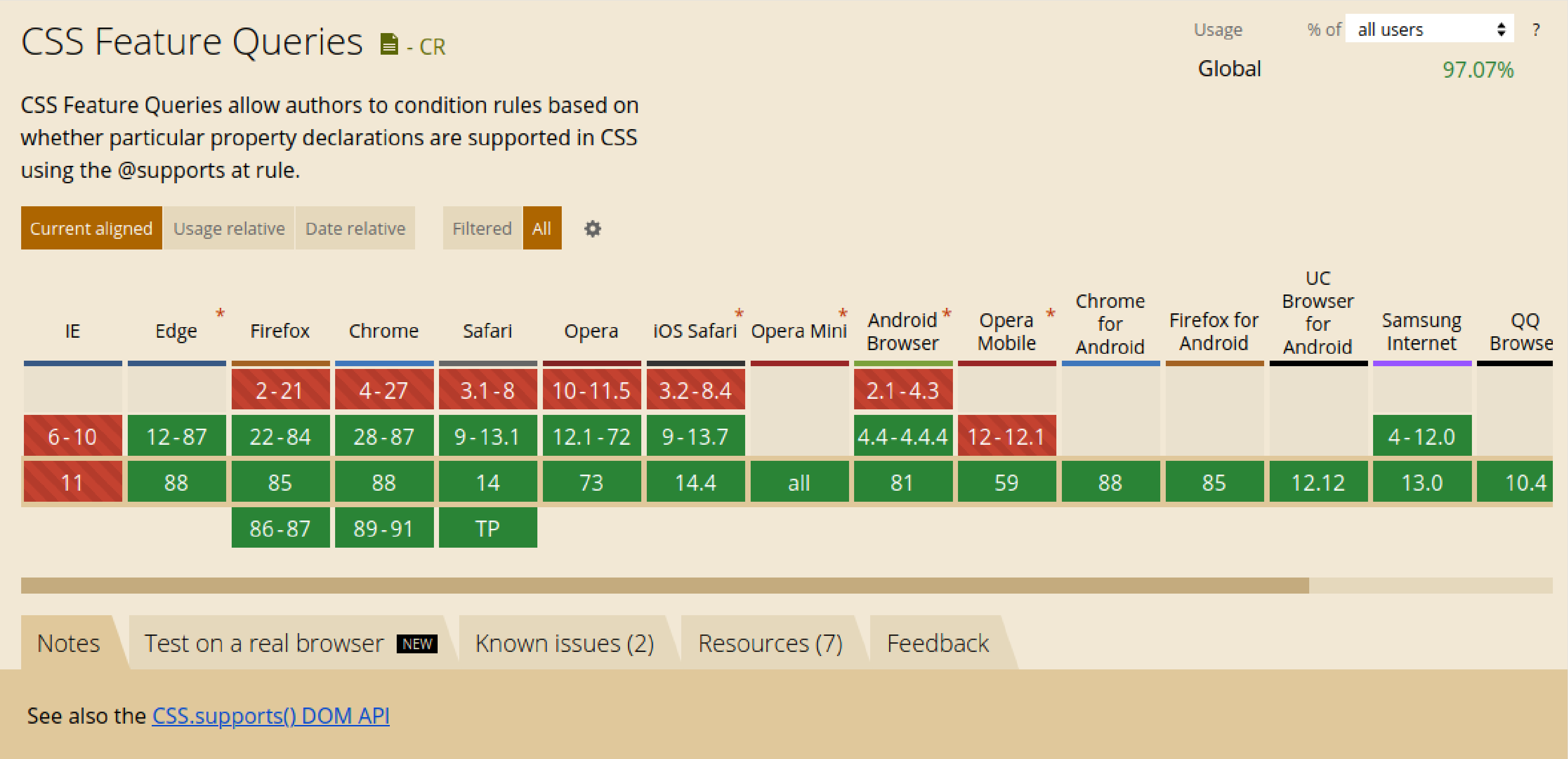
4. Which browsers use it?

5. What are vendor prefixes?

6. Does @supports replace them?

Which browsers use it?

- @supports (condition) { … } at **97%** support



CSS Feature Queries - CR

Usage % of all users ?

Global 97.07%

CSS Feature Queries allow authors to condition rules based on whether particular property declarations are supported in CSS using the @supports at rule.

IE	Edge *	Firefox	Chrome	Safari	Opera	iOS Safari *	Opera Mini *	Android Browser *	Opera Mobile *	Chrome for Android	Firefox for Android	UC Browser for Android	Samsung Internet	QQ Browse
		2-21	4-27	3.1-8	10-11.5	3.2-8.4		2.1-4.3						
6-10	12-87	22-84	28-87	9-13.1	12.1-72	9-13.7		4.4-4.4.4	12-12.1				4-12.0	
11	88	85	88	14	73	14.4	all	81	59	88	85	12.12	13.0	10.4
		86-87	89-91	TP										

See also the CSS.supports() DOM API

@supports

1. What is @supports?

2. What is the syntax?

3. When and why was it created?

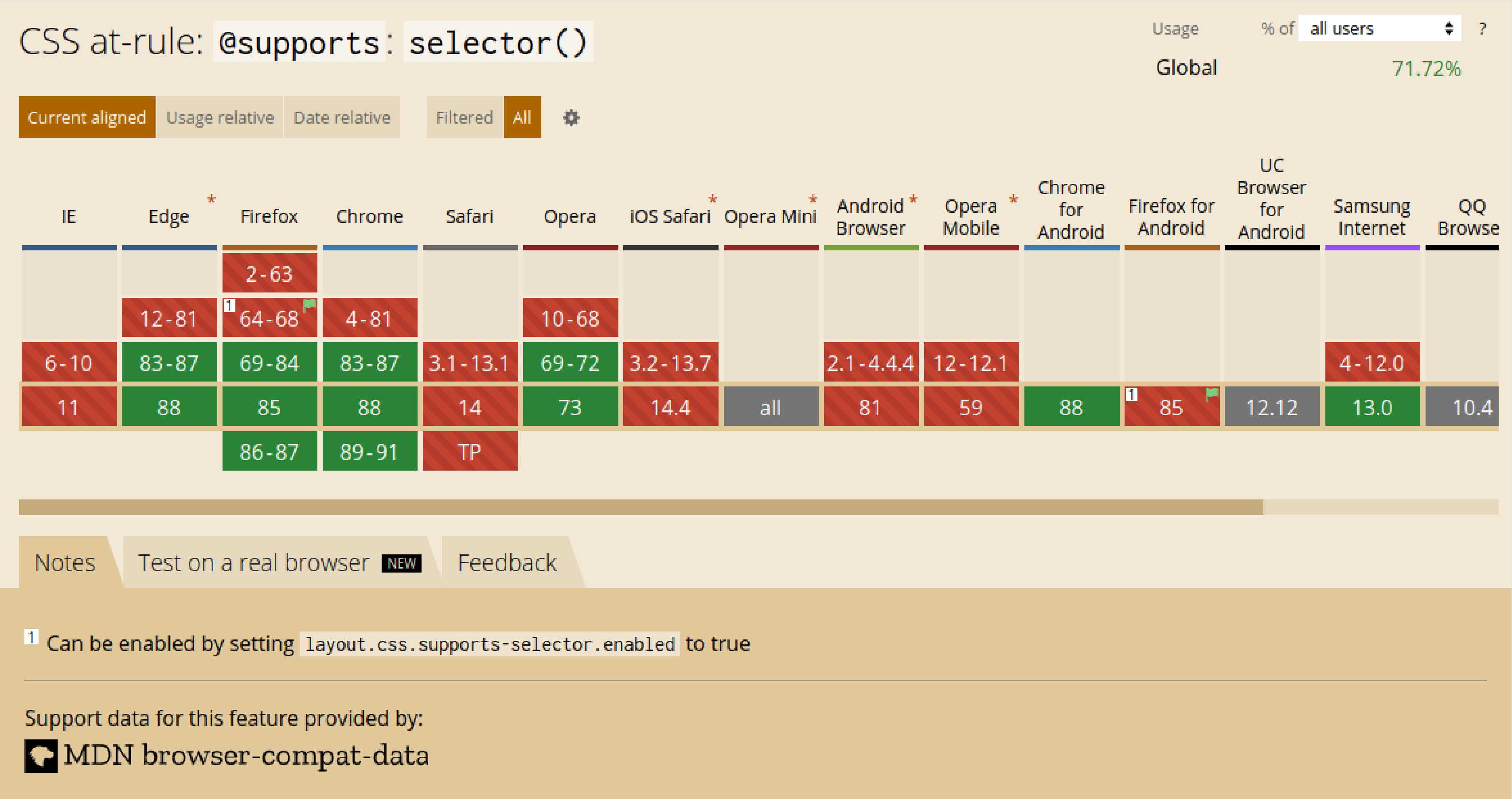
4. Which browsers use it?

5. What are vendor prefixes?

6. Does @supports replace them?

Which browsers use it?

- @supports selector(type) { … } at **72%** support



CSS at-rule: `@supports`: `selector()`

Usage % of all users ?

Global 71.72%

IE	Edge *	Firefox	Chrome	Safari	Opera	iOS Safari *	Opera Mini *	Android Browser *	Opera Mobile *	Chrome for Android	Firefox for Android	UC Browser for Android	Samsung Internet	QQ Browse
		2-63												
	12-81	[1] 64-68	4-81		10-68									
6-10	83-87	69-84	83-87	3.1-13.1	69-72	3.2-13.7		2.1-4.4.4	12-12.1				4-12.0	
11	88	85	88	14	73	14.4	all	81	59	88	[1] 85	12.12	13.0	10.4
		86-87	89-91	TP										

[1] Can be enabled by setting `layout.css.supports-selector.enabled` to true

Support data for this feature provided by:
MDN browser-compat-data

@supports

1. What is @supports?

2. What is the syntax?

3. When and why was it created?

4. Which browsers use it?

5. What are vendor prefixes?

6. Does @supports replace them?

What are vendor prefixes?

- Vendor prefixes are experimental or non-standard CSS properties and JavaScript APIs
- They are used during the development and finalisation phases of new feature implementation
- Most familiar will be the commonly seen browser prefixed values for Firefox (**-moz-**) and Chrome (**-webkit-**)
- Best practice is to list the vendor prefixed declarations first, and then the standardised W3C release version

@supports

1. What is @supports?

2. What is the syntax?

3. When and why was it created?

4. Which browsers use it?

5. What are vendor prefixes?

6. Does @supports replace them?

What are vendor prefixes?

```css
/* example: typical ordering */
div.example {
   -webkit-border-radius: 6px;
   -moz-border-radius: 6px;
   -ms-border-radius: 6px;
   -o-border-radius: 6px;
   border-radius: 6px;
}
```

What are vendor prefixes?

1. What is @supports?
2. What is the syntax?
3. When and why was it created?
4. Which browsers use it?
5. What are vendor prefixes?
6. Does @supports replace them?

- This ordering ensures that the release version takes precedence over the experimental version
- This prevents any differences between the two from causing the release version to be superseded in use
- They were supposed to allow developers to test features that are not officially supported and not relied upon in production
- However, web developers have been using them on public sites, so vendors are considering discontinuing their use

1. What is @supports?
2. What is the syntax?
3. When and why was it created?
4. Which browsers use it?
5. What are vendor prefixes?
6. Does @supports replace them?

Does @support replace them?

- Not necessarily: while @supports could reduce their use, they serve very different purposes.
- They can be used together to test if the final implementation of a feature is complete, and if not, apply the prefixed versions
- This is most useful when there are many prefixes which can be isolated for future removal to lighten code

@supports

1. What is @supports?

2. What is the syntax?

3. When and why was it created?

4. Which browsers use it?

5. What are vendor prefixes?

6. Does @supports replace them?

Does @supports replace them?

```
/* case 1: assume supported */
div.example {
   border-radius: 6px;
}

/* case 2: vendor fallbacks */
@supports not (border-radius) {
   div.example {
      -webkit-border-radius: 6px;
      -moz-border-radius: 6px;
      -ms-border-radius: 6px;
      -o-border-radius: 6px;
   }
}
```

@supports

1. What is @supports?

2. What is the syntax?

3. When and why was it created?

4. Which browsers use it?

5. What are vendor prefixes?

6. Does @supports replace them?

The End!

Questions?