CSS IS AWESOME!

$ whoami

Igor Laborie

Expert Java & Web, 

🐦 @ilaborie

✉️ igor@monkeypatch.io

⚠️ *Je ne suis pas un designer*

The Rule of Least Power

> When designing computer systems, one is often faced with a choice between using a more or less powerful language for publishing information, for expressing constraints, or for solving some problem. This finding explores tradeoffs relating the choice of language to reusability of information. The "Rule of Least Power" suggests **choosing the least powerful language suitable** for a given purpose.

Règles du jeu

1. Texte
2. HTML (sémantique)
3. CSS (layout, style, animations simples)
4. SVG (formes et animations complexes)
5. JavaScript, WebAssembly (gestion d'états, appel backend, calculs)

⚠️*... mais il y a toujours de bonnes raisons pour ne pas suivre ces règles*

Le CSS c'est vaste

- Selectors
- Box model
- Float
- Media Query
- Animations
- Gradients
- Responsive Design
- Media
- Variables
- Colors
- Shapes
- ...

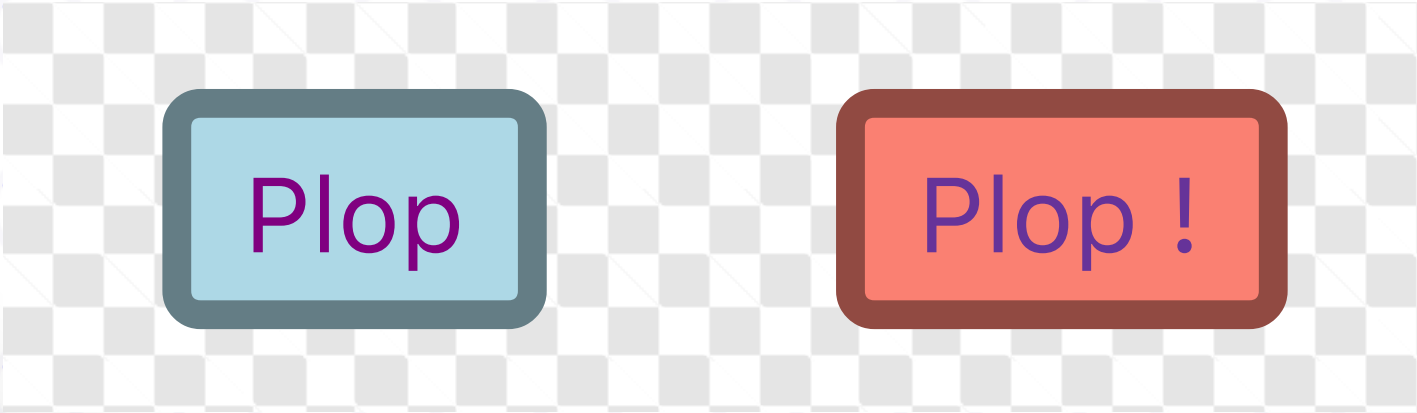
Plan

I. Utiliser un pré-processeur ?
II. Unités
III. Flexbox et Grid
IV. Pseudo éléments
V. Animations
VI. Pseudo classes d'état
VII. Conclusion

Utiliser un pré-processeur ?

Bordure des boutons


```css
button {
  background: lightblue;
  color: purple;
  /*border: medium solid currentColor;*/
  border: medium solid rgba(0,0,0,.42);
}
button.danger {
  background: salmon;
  color: rebeccapurple;
}
```



Alors utilise-t-on un pré-processeurs ?

Oui, mais privilégiez:

- le CSS
- les post-processeurs

- ✻ `currentColor`
- 🦊 `background-origin`
- ***w3c*** CSS Variables (aka Custom Properties)
- ***w3c*** CSS Color Module Level 4

Compatibilité

Navigateurs, usage ≥ 0,4% au Maroc

	32,7 %	29,6 %	6,3 %	5,0 %	4,7 %	1,8 %	1,4 %	1,4 %	0,7 %	0,6 %
CSS currentColor value	43 49 55 58 59 60	61	52 54 55	11 10.0-10.2 10.3	4.2-4.3 4.4 4.4.3-4.4.4	46 47	all	8 11	10.1	14
CSS3 Background-image options	43 49 55 58 59 60	61	52 54 55	11 10.0-10.2 10.3	4.2-4.3 4.4 4.4.3-4.4.4	46 47	all	8 11	10.1	14
CSS Variables (Custom Properties)	43 49 55 58 59 60	61	52 54 55	11 10.0-10.2 10.3	4.2-4.3 4.4 4.4.3-4.4.4	46 47	all	8 11	10.1	14

Unités

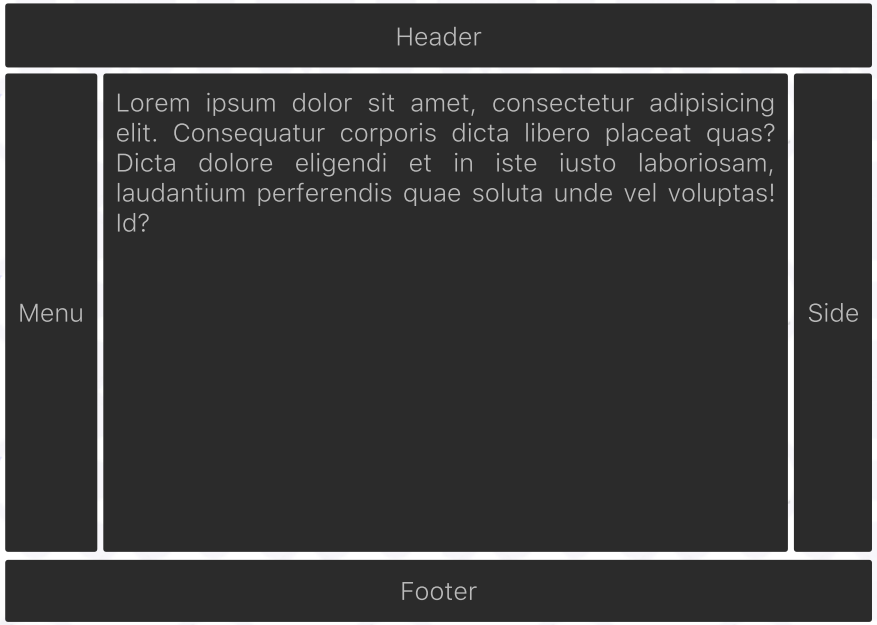
Une histoire d'unités CSS

🎨 CommitStrip

Les unités de longueur

px, cm, pt, ...	longueurs absolues (mesure physique)
em, rem	fonction de la `font-size`
ex, ch	hauteur d'un `x`, largeur d'un `0`
vh, vw	(100vh, 100vw) = (hauteur, largeur) du *viewport*
vmin, vmax	min(1vh, 1vw), max(1vh, 1vw)

Holy Grail avec **calc**




```
<body>
  <header>Header</header>
  <div>
    <nav>Menu</nav>
    <main>Content</main>
    <aside>Side</aside>
  </div>
  <footer>Footer</footer>
</body>
```

Bilan unités

- 🦊 Unités
- *w3c* Truc et astuces
- 🦊 `calc`
- ✻ Fun with Viewport Units

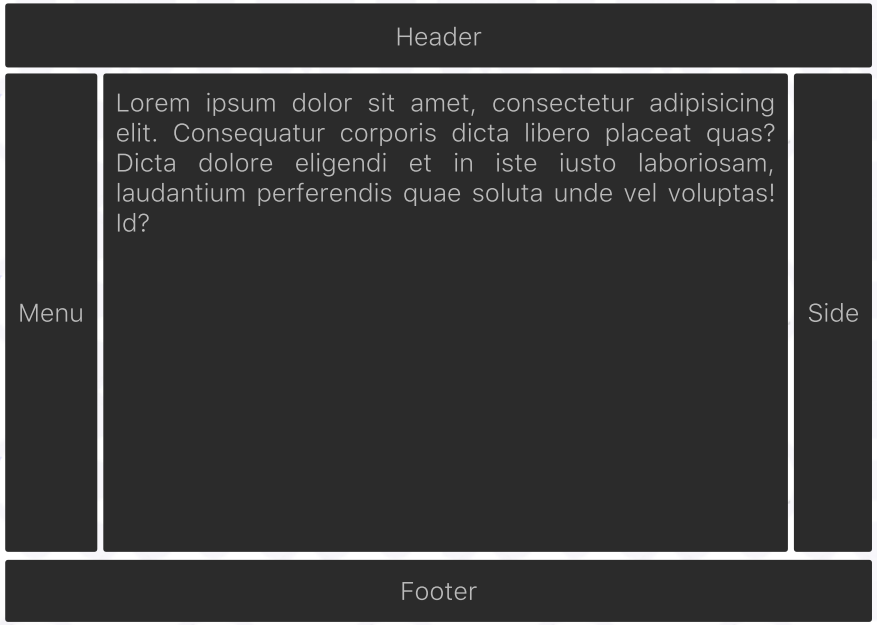
Compatibilité

Navigateurs, usage ≥ 0,4% au Maroc

	32,7 %	29,6 %	6,3 %	5,0 %	4,7 %	1,8 %	1,4 %	1,4 %	0,7 %	0,6 %
rem (root em) units	43 49 55 58 59 60	61	52 54 55	11 10.0-10.2 10.3	4.2-4.3 4.4 4.4.3-4.4.4	46 47	all	8 11	10.1	14
Viewport units: vw, vh, vmin, vmax	43 49 55 58 59 60	61	52 54 55	11 10.0-10.2 10.3	4.2-4.3 4.4 4.4.3-4.4.4	46 47	all	8 11	10.1	14
calc() as CSS unit value	43 49 55 58 59 60	61	52 54 55	11 10.0-10.2 10.3	4.2-4.3 4.4 4.4.3-4.4.4	46 47	all	8 11	10.1	14

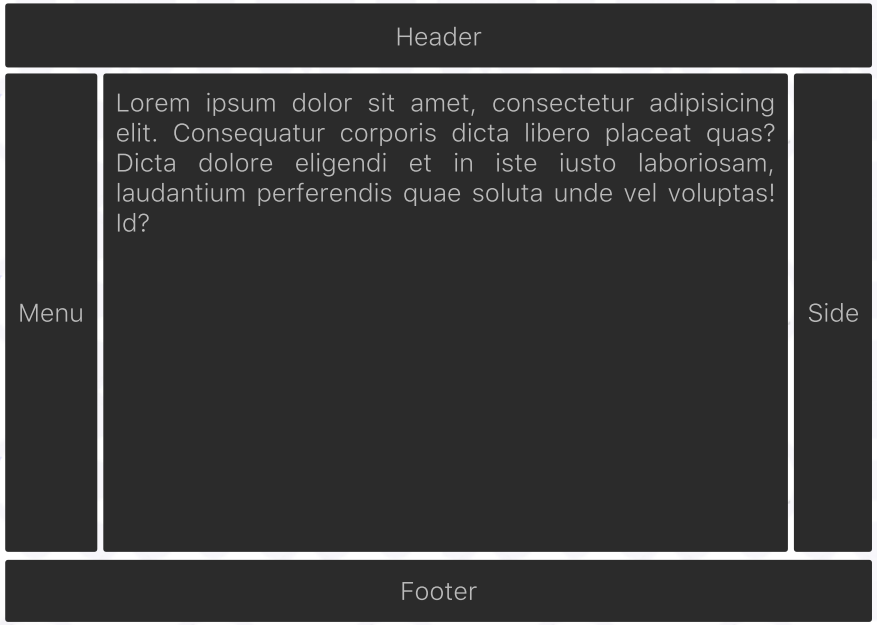
Flexbox et Grid

Holy Grail avec **flexbox**




```
<body>
  <header>Header</header>
  <div>
    <nav>Menu</nav>
    <main>Content</main>
    <aside>Side</aside>
  </div>
  <footer>Footer</footer>
</body>
```

Holy Grail avec **grid**




```
<body>
  <header>Header</header>
  <div>
    <nav>Menu</nav>
    <main>Content</main>
    <aside>Side</aside>
  </div>
  <footer>Footer</footer>
</body>
```

Bilan Flexbox & Grid

Flexbox

- 📽️ Flexbox, et le CSS redevient fun ! (Hubert SABLONNIÈRE)
- Solved by Flexbox
- 🐸 Flexbox Froggy

Grid

- 🦊 `@supports`
- # Grid by examples
- 📽️ CSS Grid Changes Everything (About Web Layouts) by Morten Rand-Hendriksen
- 🥕 Grid Garden

Compatibilité

Navigateurs, usage ≥ 0,4% au Maroc

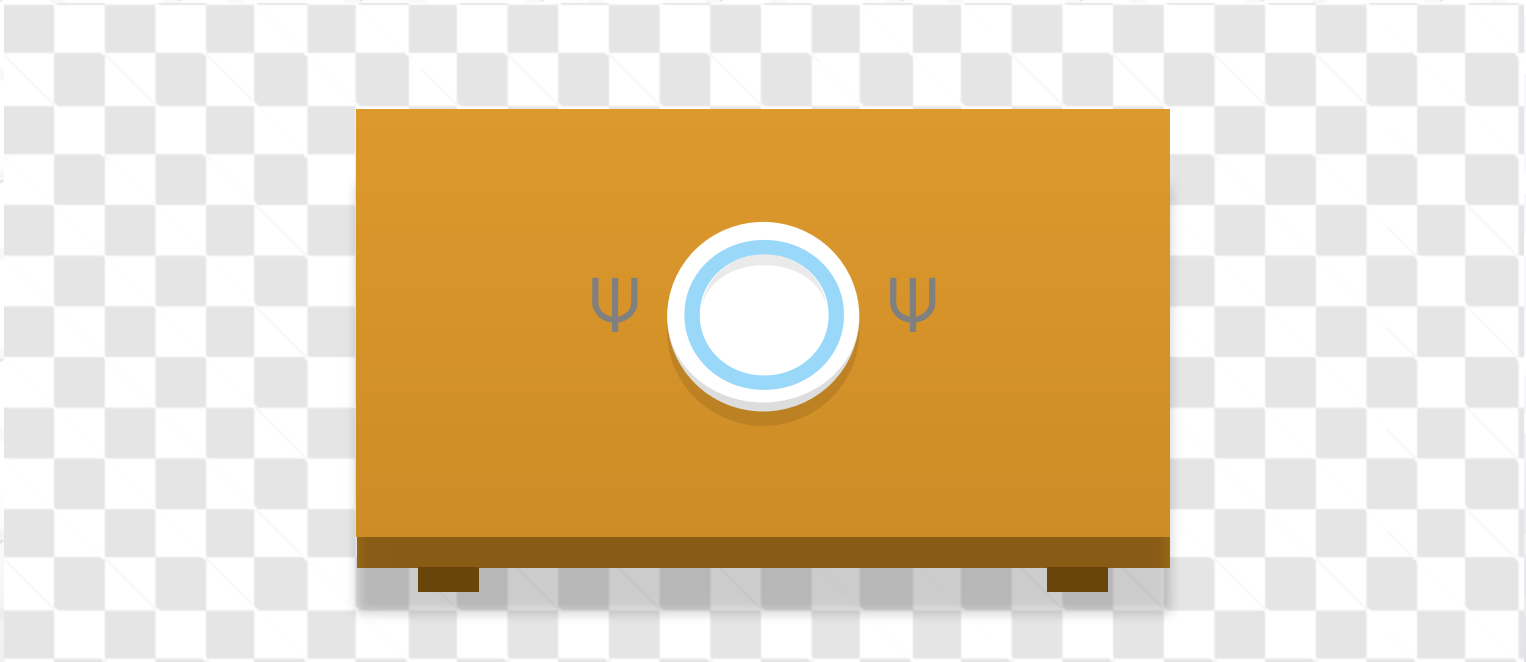
	32,7 %	29,6 %	6,3 %	5,0 %	4,7 %	1,8 %	1,4 %	1,4 %	0,7 %	0,6 %
CSS Flexible Box Layout Module	43 49 55 58 59 60	61	52 54 55	11 10.0-10.2 10.3	4.2-4.3 4.4 4.4.3-4.4.4	46 47	all	8 11	10.1	14
CSS Grid Layout	43 49 55 58 59 60	61	52 54 55	11 10.0-10.2 10.3	4.2-4.3 4.4 4.4.3-4.4.4	46 47	all	8 11	10.1	14
CSS Feature Queries	43 49 55 58 59 60	61	52 54 55	11 10.0-10.2 10.3	4.2-4.3 4.4 4.4.3-4.4.4	46 47	all	8 11	10.1	14

Pseudo éléments

Le dîner d'un philosophe


```html
<div class="table">
  <div class="plate"></div>
</div>
```

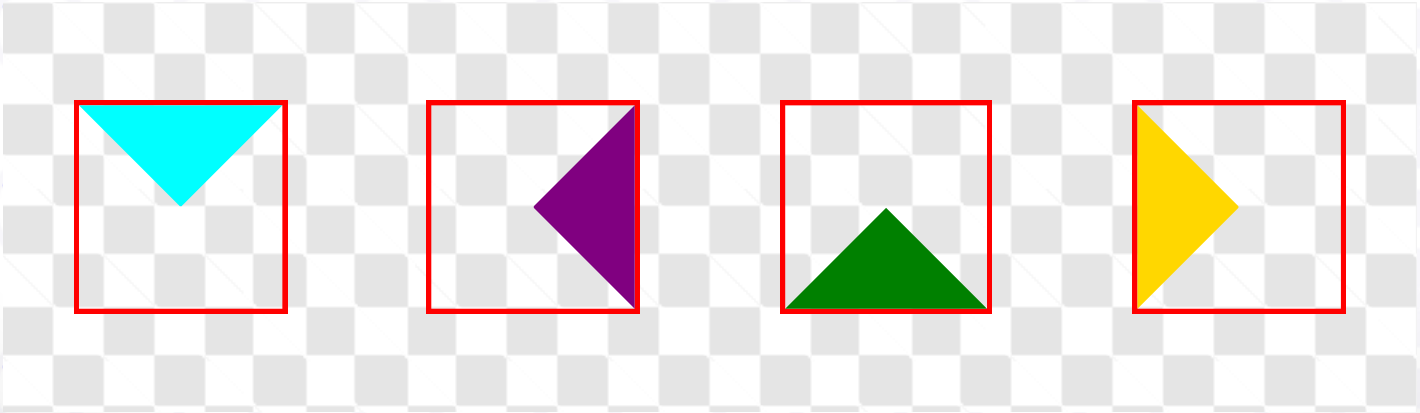
```css
.table::before, .table::after {
  color: gray;
  font-size: 2rem;
  content: 'ⴲ';
  transform: rotate(180deg);
}
```



Triangle avec des bordures


```css
div.top, div.right, div.bottom, div.left {
  border: 2em solid transparent;
  display: inline-block;
  box-shadow: 0 0 0 .1em red;
}

div.top { border-top-color: cyan; }
div.right { border-right-color: purple; }
div.bottom { border-bottom-color: green; }
div.left { border-left-color: gold; }
```



Info-bulle


```css
.popover {
  position: relative;
  background: teal;
}
.popover::before {
  position: absolute;
  z-index: -1;
  content: '';
  top: 1.25em; left: 1em;
  border: .8em solid transparent;
  border-top-color: teal;
  transform: skew(-30deg);
}
```

Hello Devoxx Morocco !

Bilan pseudo éléments

- w3c The :before and :after pseudo-elements
- mais aussi `::first-letter`, `::first-line`, `::selection`, `::backdrop`
- S An Ultimate Guide To CSS Pseudo-Classes And Pseudo-Elements

⚠️ `::before` et `::after` ne marchent pas sur `input`, `img`, `iframe` (pas encore spécifié)

- Table et assiette de CSS Diner
- Ⓦ Dîner des philosophes

Compatibilité

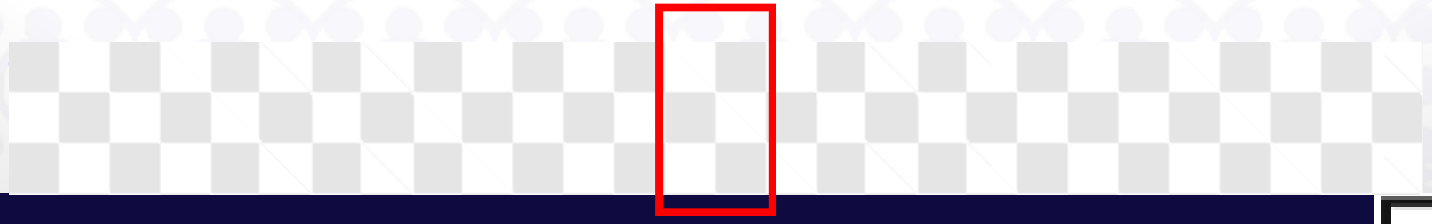
Navigateurs, usage ≥ 0,4% au Maroc

	32,7 %	29,6 %	6,3 %	5,0 %	4,7 %	1,8 %	1,4 %	1,4 %	0,7 %	0,6 %
CSS Generated content for pseudo-elements	43 49 55 58 59 60	61	52 54 55	11 10.0-10.2 10.3	4.2-4.3 4.4 4.4.3-4.4.4	46 47	all	8 11	10.1	14
CSS3 2D Transforms	43 49 55 58 59 60	61	52 54 55	11 10.0-10.2 10.3	4.2-4.3 4.4 4.4.3-4.4.4	46 47	all	8 11	10.1	14

Animations

Texte de chargement


```
.loader {
  display: inline-block;
  white-space: pre;
  height: 1.3em; margin-top: -.3rem;
  line-height: 1.5;
  overflow: hidden;
}
.loader::before {
  display: inline-table;
  content: '⁘\a ⁘\a ⁙\a ⁙\a ⁖\a ⁖\a ⁙\a ⁙\a ⁘';
  animation: spin 1s steps(10, end) infinite;
}
@keyframes spin {
  to { transform: translateY(-15em); }
}
```



Dessiner


```css
.editable svg path {
  stroke: purple;
  stroke-width: 1em;
  fill: none;
  stroke-dasharray: 4700;
  stroke-dashoffset: 4700;
  animation: draw 2s linear infinite;
}
@keyframes draw {
  to { stroke-dashoffset: 0; }
}
```

Bilan animations

- 🦊 Utiliser les animations CSS
- 💨 Text spinners
- 💨 CSS only loaders
- Animate.css
- ✳ How SVG Line Animation Works
- 💨 Animated line drawing in SVG
- 💨 CSS triggers

Compatibilité

Navigateurs, usage ≥ 0,4% au Maroc

	32,7 %	29,6 %	6,3 %	5,0 %	4,7 %	1,8 %	1,4 %	1,4 %	0,7 %	0,6 %
CSS Animation	43 49 55 58 59 60	61	52 54 55	11 10.0-10.2 10.3	4.2-4.3 4.4 4.4.3-4.4.4	46 47	all	8 11	10.1	14
SVG (basic support)	43 49 55 58 59 60	61	52 54 55	11 10.0-10.2 10.3	4.2-4.3 4.4 4.4.3-4.4.4	46 47	all	8 11	10.1	14

Pseudo classes d'état

Usage des info-bulles

Pseudo états

- :hover
- :focus
- :visited
- :checked
- :valid
- :invalid
- :empty
- :target
- …

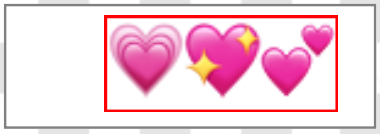
Checkbox Hack


```css
.editable input[type=checkbox] + label::before {
  content: 'Click if you like it';
}

.editable input[type=checkbox]:checked + label::before {
  content: '💗💖💕';
}

fieldset input[type=checkbox] { opacity: 0; }
```

> The science of operations, as derived from mathematics more especially, is a science of itself, and has its own abstract truth and value.
>
> ⟿ *Ada Lovelace*



Switch


```css
.switch + label {
  display: block;
  position: relative;
  padding: .1em;
  width: 2em;
  height: 1em;
  background-color: #ccc;
  border-radius: 1em;
  border: medium solid #444;
  transition: 0.4s;
}

.switch:checked + label {
  background-color: green;
}
```

```css
.switch + label::before {
  display: block;
  position: absolute;
  content: '';
  top: 0.1em;
  left: 0.1em;
  height: 1em;
  width: 1em;
  background-color: #fff;
  border-radius: 50%;
  transition: all 0.25s;
}

.switch:checked + label::before {
  transform: translateX(1em);
}
```

Hiding Content for Accessibility

```
.panel input[type=checkbox] {
  position: fixed;
  left: -100vmax;
}
```

Apollo 11

> The computer (or rather the software in it) was smart enough to recognize that it was being asked to perform more tasks than it should be performing. It then sent out an alarm, which meant to the astronaut, I'm overloaded with more tasks than I should be doing at this time and I'm going to keep only the more important tasks; i.e., the ones needed for landing ... Actually, the computer was programmed to do more than recognize error conditions. A complete set of recovery programs was incorporated into the software. The software's action, in this case, was to eliminate lower priority tasks and re-establish the more important ones ... If the computer hadn't recognized this problem and taken recovery action, I doubt if Apollo 11 would have been the successful moon landing it was.[26]
>
> *Letter from Margaret H. Hamilton, Director of Apollo Flight Computer Programming MIT Draper Laboratory, Cambridge, Massachusetts[27], titled "Computer Got Loaded", published in Datamation, March 1, 1971*

Principe pour les onglets


```
<div class="tabs">
  <input type="radio" name="tab" id="home" checked>
  <input type="radio" name="tab" id="projects">
  <input type="radio" name="tab" id="about">
  <nav>
    <label for="home">Home</label>
    <label for="projects">Projects</label>
    <label for="about">About</label>
  </nav>
  <div data-for="home">Home page</div>
  <div data-for="projects">Projects page</div>
  <div data-for="about">About page</div>
</div>
```

Démo des onglets

Compatibilité

Navigateurs, usage ≥ 0,4% au Maroc

	32,7 %	29,6 %	6,3 %	5,0 %	4,7 %	1,8 %	1,4 %	1,4 %	0,7 %	0,6 %
CSS3 selectors	43 49 55 58 59 60	61	52 54 55	11 10.0-10.2 10.3	4.2-4.3 4.4 4.4.3-4.4.4	46 47	all	8 11	10.1	14
Form validation	43 49 55 58 59 60	61	52 54 55	11 10.0-10.2 10.3	4.2-4.3 4.4 4.4.3-4.4.4	46 47	all	8 11	10.1	14
CSS3 3D Transforms	43 49 55 58 59 60	61	52 54 55	11 10.0-10.2 10.3	4.2-4.3 4.4 4.4.3-4.4.4	46 47	all	8 11	10.1	14

Conclusion

Bilan

1. Utilisez du CSS pour simpifier le code
2. Utilisez intelligemment les pre/post-processeurs
3. HTML, SVG are Awesome !
4. JavaScript, TypeScript can be Awesome !

👉 Traitez le CSS comme du code

1. Revue de code
2. DRY
3. Clean Code
4. Single Responsibility Principle
5. ...

- les slides en HTML
- les slides en PDF
- le code
- 🍌 Blog: 'Making Of'

Pour apprendre

- `(Ctrl|⌘) + Shift + i`
- ➳ CSS Secrets by Lea Verou
- 🦊 CSS sur MDN , ➳ The A11Y Project
- ➳ CodePen , ➳ JSFiddle , ➳ Dabblet ,...
- ✻ CSS Tricks , 𝓢 Smashing Magazine
- CSS Flags , ➳ A Single Div

CSS
is
Awesome!