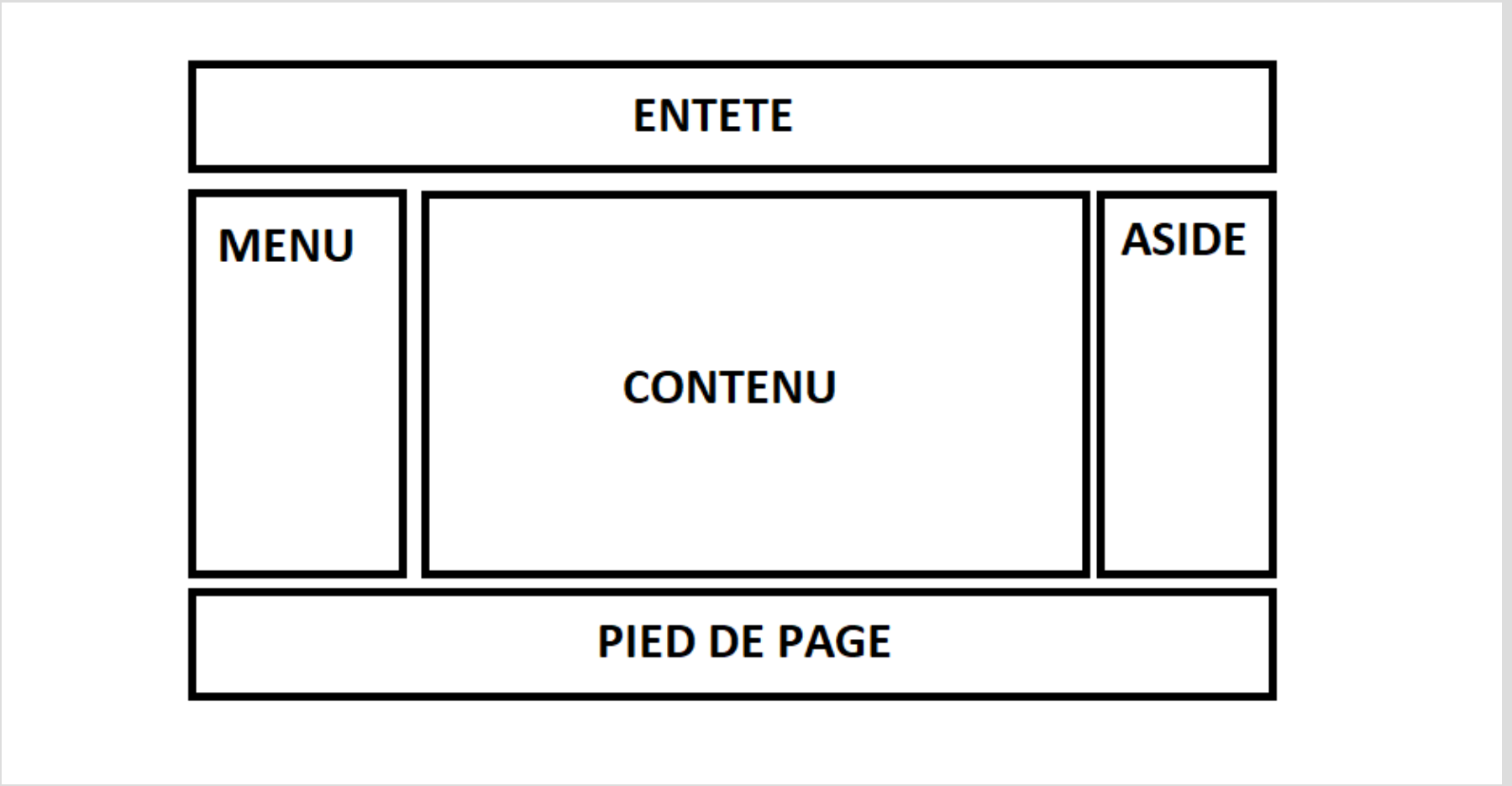
CSS3 – Grid Layout

- Mise en page : cas pratique
- Définir la grille
- Dimensionner la grille
- Méthode 1 : définir les lignes de grille
- Placer les éléments
- Méthode 2 : nommer les lignes
- Définir les gouttières
- Méthode 3 : définir et nommer les zones

Exemple de mise en page

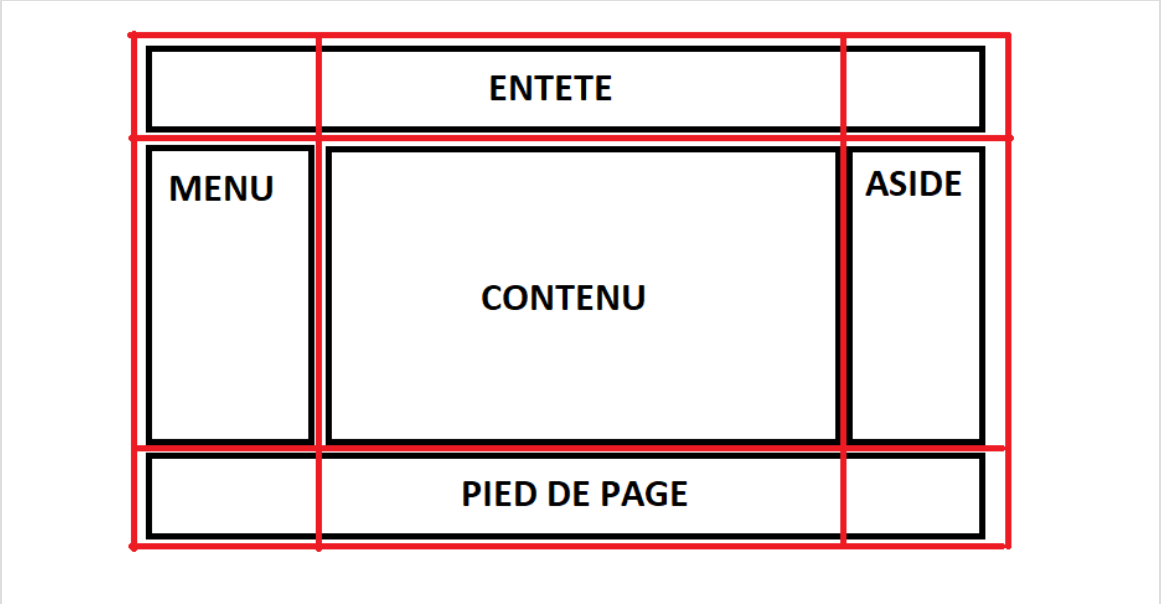
- Comment organiser le contenu (sur ce schéma) ?



Mise en page : cas pratique

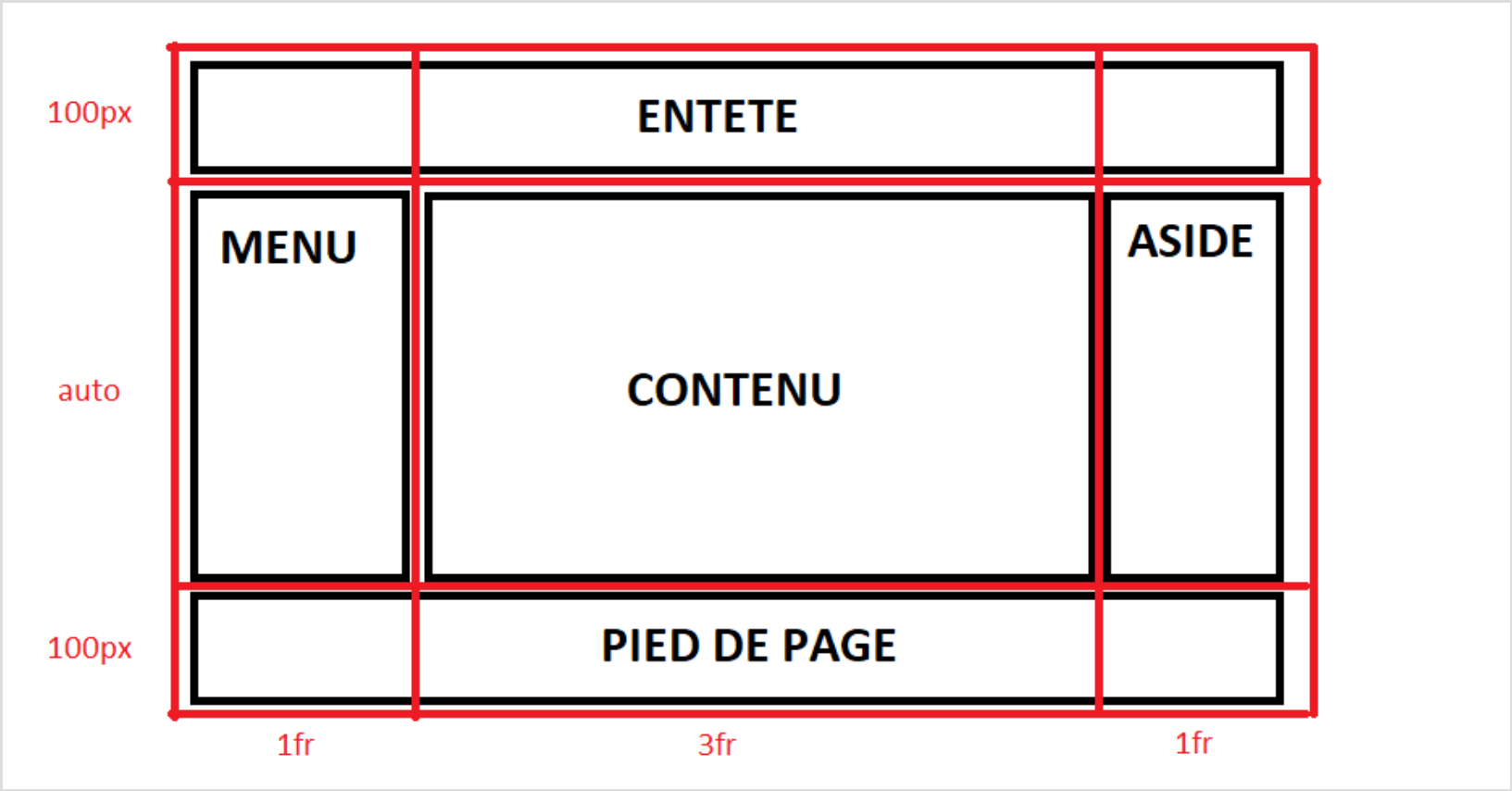
- Mettre la mise en page dans une grille Comment ?
- Définir la grille (nombre de rangées et colonnes, dimensions). Comment ?
- Placer les éléments dans la grille en fonction des pistes, lignes, cellules ou zones. Comment ?
- Définir ces pistes, lignes, cellules ou zones Comment ?

Définir la grille



- **3 colonnes et 3 rangées**
  - Entête et pied de page : hauteur fixe et toute la largeur
  - Menu et Barre latérale (aside) : hauteur automatique et 1/5 de la largeur de page
  - Contenu : hauteur automatique et 1/5 de la largeur

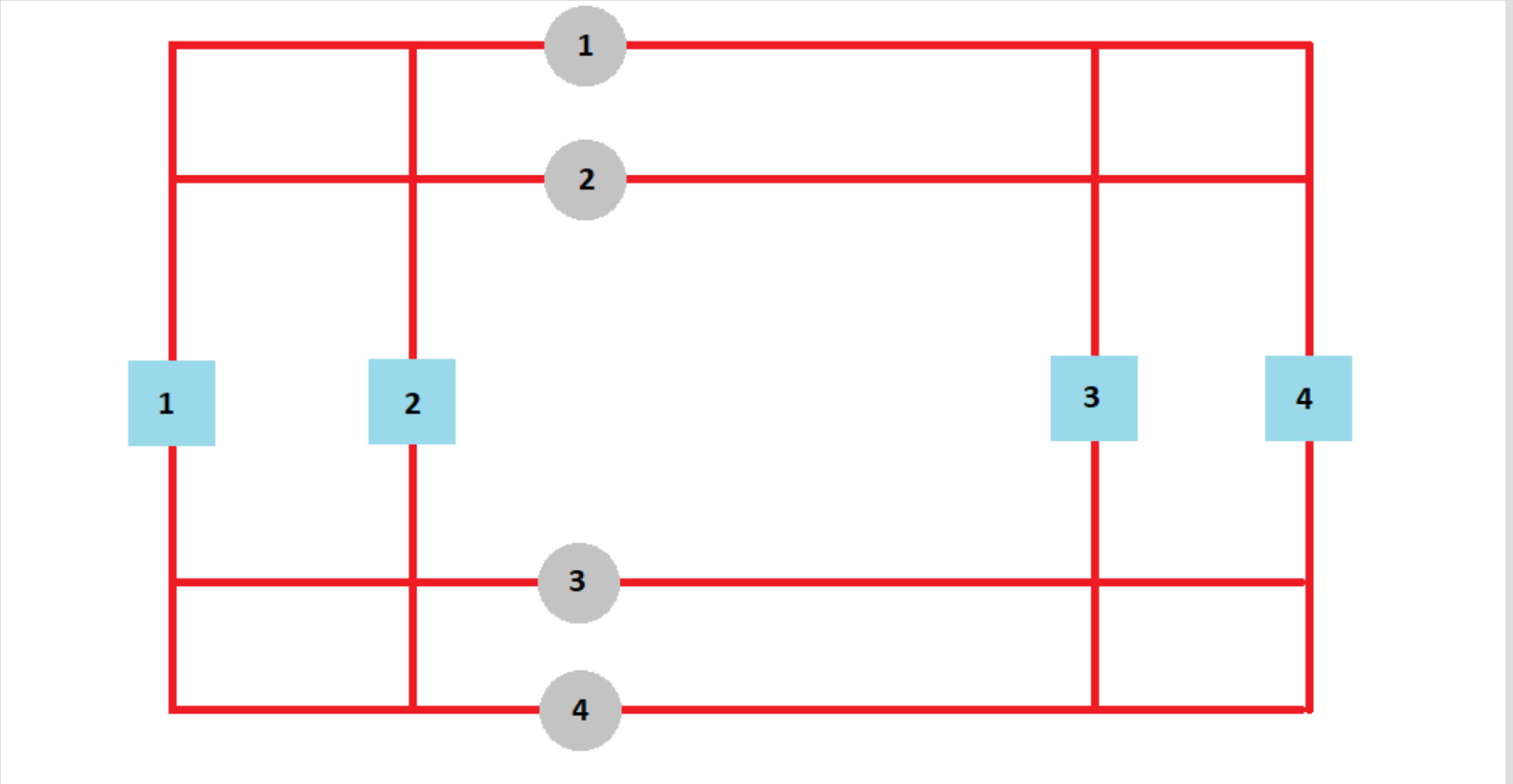
Dimensionner la grille




```
#global {
    display:grid;
    grid-template-columns: 1fr 3fr 1fr;
    grid-template-rows: 100px auto 100px;
}
```

Lignes de grille

- Chaque colonne et chaque rangée est délimitée par des **lignes numérotées à partir de 1**



Placer les éléments

- Pour **placer** les éléments selon les **lignes de la grille**, on utilise les numéros de ligne :
  - **grid-column-start** : ligne de colonne de début
  - **grid-column-end** : ligne de colonne de fin
  - **grid-row-start** : ligne de rangée de début
  - **grid-row-end** : ligne de rangée de fin
- Exemple : l'entête - colonne 1 à 4 , rangée 1 à 2
  - grid-column-start : 1; grid-column-end : 4;
  - grid-row-start : 1; grid-row-end : 2;

Mise en page – HTML

```html
<!DOCTYPE html>
<html lang="fr">
<head>
    <meta charset="UTF-8">
    <title>CSS Grid Layout</title>
    <link type="text/css" rel="stylesheet" href="grid.css" />
</head>
<body>
    <div id="global">
        <header><p>Entête</p></header>
        <nav><h2>Menu</h2></nav>
        <main>
            <h1>Contenu</h1>
            <p>Lorem ipsum dolor sit amet, consectetur
            adipisicing elit. Velit quis molestias nulla rerum
            ex, fugiat blanditiis porro dolores. Obcaecati
            quibusdam fugit velit inventore asperiores
            repudiandae maxime molestias dolores, eius ex.</p>
            <p>Assumenda in fuga consequuntur amet neque
            labore repellat quo architecto ab quod. Provident,
            quia perspiciatis nisi neque voluptates quaerat
            quae laudantium amet.</p>
        </main>
        <aside><h2>Barre latérale</h2></aside>
        <footer><p>Pied de page</p></footer>
    </div>
</body>
</html>
```

Mise en page – CSS

```css
#global {
    display:grid;
    grid-template-columns: 1fr 3fr 1fr;
    grid-template-rows: 100px auto 100px;
}

header {
    background-color: aqua;
    grid-column-start: 1;
    grid-column-end: 4;
    grid-row-start: 1;
    grid-row-end: 2;
}

nav {
    background-color: aquamarine;
    grid-column-start: 1;
    grid-column-end: 2;
    grid-row-start: 2;
    grid-row-end: 3;
}
```

```css
main {
    background-color: coral;
    grid-column-start: 2;
    grid-column-end: 3;
    grid-row-start: 2;
    grid-row-end: 3;
}

aside {
    background-color: lightgray;
    grid-column-start: 3;
    grid-column-end: 4;
    grid-row-start: 2;
    grid-row-end: 3;
}

footer {
    background-color: black;
    color:antiquewhite;
    grid-column-start: 1;
    grid-column-end: 4;
    grid-row-start: 3;
    grid-row-end: 4;
}
```

grid-column et grid-row

- Pour positionner des éléments sur base des lignes de grille, il faut remplir **beaucoup de propriétés**
- Propriétés raccourcies : **grid-column** et **grid-row**
  - grid-column-start : 1; grid-column-end : 4;
    devient **grid-column: 1 / 4;**
  - grid-row-start : 1; grid-row-end : 2;
    devient **grid-row : 1 / 2 ;**
- Si un élément n'occupe qu'une piste (les numéros se suivent), on peut omettre grid-***-end
  - grid-row-start : 1; ~~grid-row-end : 2;~~
    devient **grid-row : 1;**

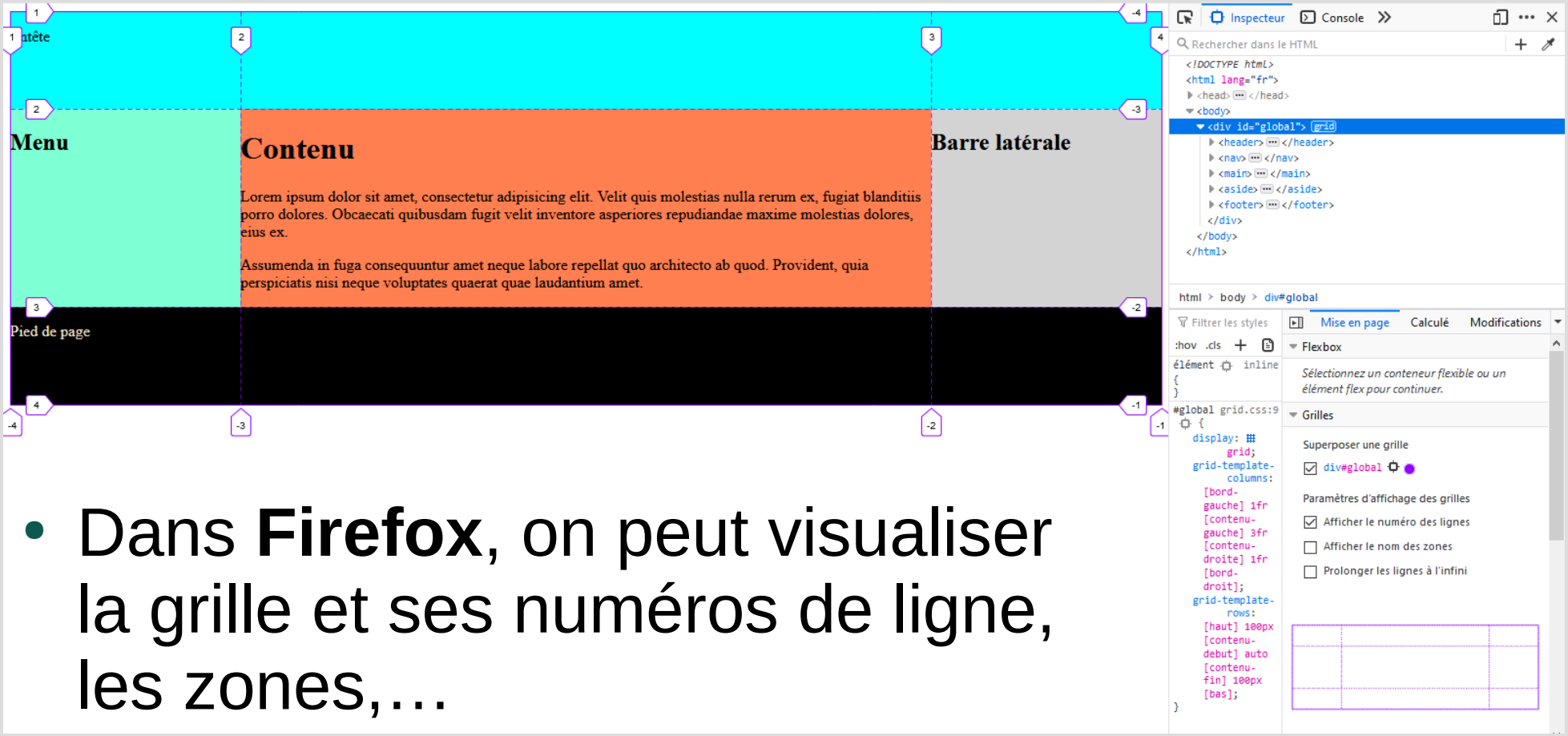
span

- Donc, pour positionner un élément, on indique les ligne de début et ligne de fin avec des **numéros**
- Il est aussi possible de définir la taille d'un élément en indiquant le numéro de la ligne de départ et le **nombre de pistes** sur lequel s'étale l'élément, avec le mot-clé **span**
- Exemple :
  - grid-column-start : 1; grid-column-end : 4;
    devient **grid-column: 1 / span 3;**

Compter à rebours

- On peut également **compter à l'envers** les lignes de grille, à partir des lignes de fin. En français, cela correspond à la colonne la plus à droite et à la ligne la plus basse.
- Pour faire référence à la **dernière ligne**, on peut utiliser la **valeur -1** , ensuite -2 fait référence à l'avant-dernière ligne,… Utile quand le **nombre de pistes** est **inconnu ou dynamique**
- Par exemple, un élément qui occupe toute la largeur aura **grid-column : 1 / -1;**

Firefox et CSS Grid Layout

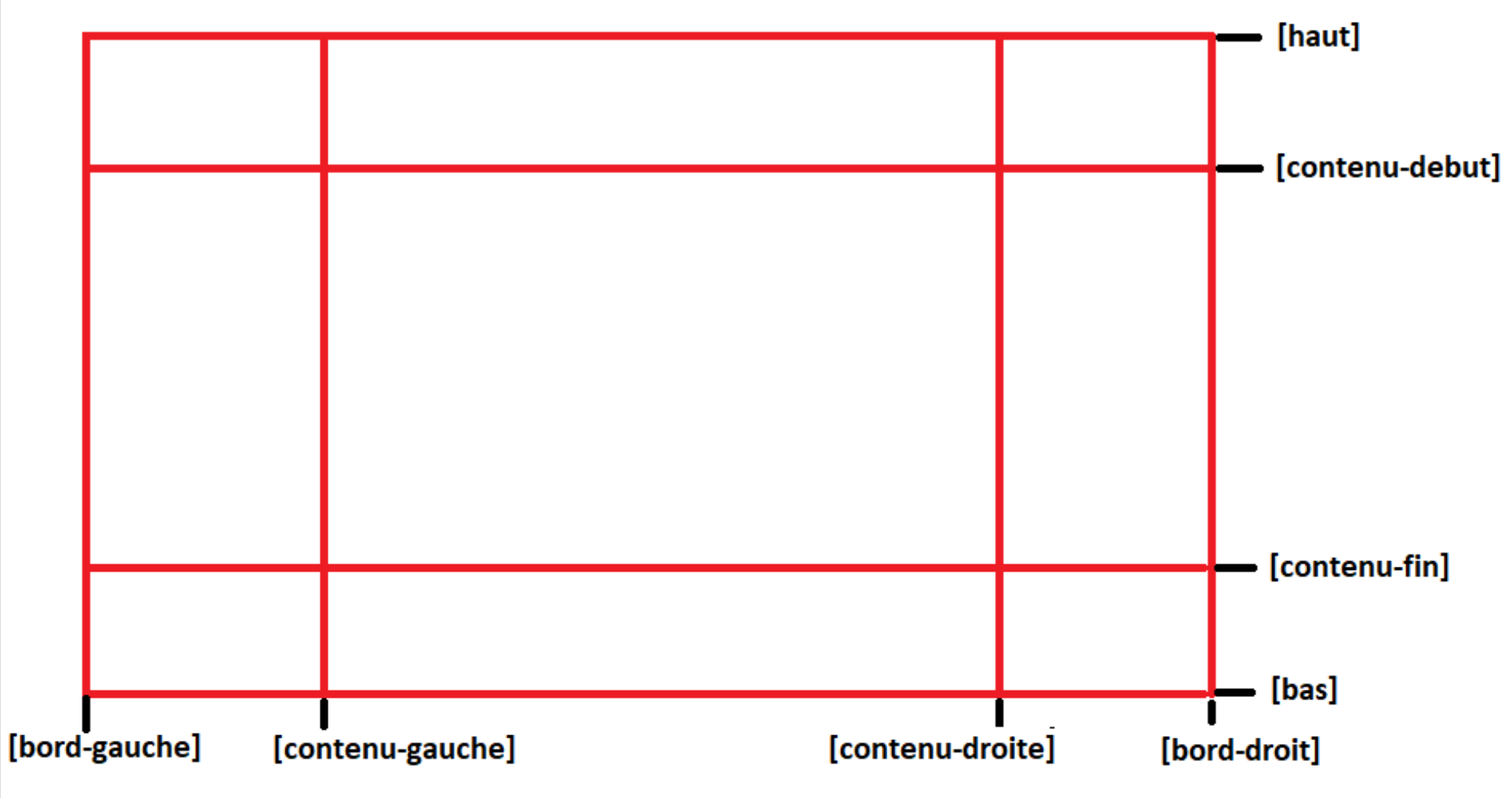


- Dans **Firefox**, on peut visualiser la grille et ses numéros de ligne, les zones,…
- On doit activer le conteneur **grid**

Nommer les lignes

- Lorsqu'on définit une grille (avec grid-template-rows et grid-template-columns) on peut **donner des noms aux lignes** au lieu d’utiliser les numéros de ligne
- Valable pour toutes ou seulement quelques unes (par exemple, celles qui sont importantes).
- On nomme les lignes **entre crochet**s. Ces noms peuvent être n'importe quelle valeur.
- On peut alors utiliser **soit le nom soit le numéro** de ligne afin de placer les éléments

Nommer les lignes de grille



Lignes nommées

```
#global {
    display:grid;
    grid-template-columns: 1fr 3fr 1fr;
    grid-template-rows: 100px auto 100px;
}

header {
    background-color: aqua;
    grid-column-start: 1;
    grid-column-end: 4;
    grid-row-start: 1;
    grid-row-end: 2;
}
```

```
#global {
    display:grid;
    grid-template-columns: [bord-gauche] 1fr [contenu-
    gauche] 3fr [contenu-droite] 1fr [bord-droit];
    grid-template-rows: [haut] 100px [contenu-debut] auto
    [contenu-fin] 100px [bas];
}

header {
    background-color: aqua;
    grid-column: bord-gauche / bord-droit;
    grid-row : haut / contenu-debut;
}
```

Ajouter des gouttières

- Pour aérer la présentation sans devoir manipuler les margin et padding, on peut créer des espaces avec les **gouttières**
  - **column-gap** : 10px;
  - **row-gap** : 15px;
- La propriété **raccourcie** reprenant les 2 valeurs
  - **gap** : 15px 10px;

Définir des zones

- On peut aussi définir la position des éléments selon des **zones** avec la propriété **grid-area**
- **grid-area** est également un raccourci de propriétés dans l'ordre suivant :
  - grid-row-start
  - grid-column-start
  - grid-row-end
  - grid-column-end
- Exemple : l'entête - colonne 1 à 4 et rangée 1 à 2
  - **grid-area : 1 / 1 / 2 / 4 ;**

Nommer les zones

- Autre manière d’utiliser les **zones**, on peut les **nommer** et ensuite **définir leurs positions**
- On utilise **grid-area** pour **nommer une zone**
- On définit la position de chaque zone dans le conteneur de la grille en utilisant le **nom de la zone et sa position** avec la propriété **grid-template-areas**
- NB : avec **Firefox**, on peut visualiser ces zones

Zones nommées – HTML

```html
<!DOCTYPE html>
<html lang="fr">
<head>
    <meta charset="UTF-8">
    <title>CSS Grid Layout - 2</title>
    <link type="text/css" rel="stylesheet" href="grid2.css" />
</head>
<body>
    <div id="global">
        <header><p>Entête</p></header>
        <nav><h2>Menu</h2></nav>
        <main>
            <h1>Contenu</h1>
            <p>Lorem ipsum dolor sit amet, consectetur
            adipisicing elit. Velit quis molestias nulla rerum
            ex, fugiat blanditiis porro dolores. Obcaecati
            quibusdam fugit velit inventore asperiores
            repudiandae maxime molestias dolores, eius ex.</p>
            <p>Assumenda in fuga consequuntur amet neque
            labore repellat quo architecto ab quod. Provident,
            quia perspiciatis nisi neque voluptates quaerat
            quae laudantium amet.</p>
        </main>
        <aside><h2>Barre latérale</h2></aside>
        <footer><p>Pied de page</p></footer>
    </div>
</body>
</html>
```

Zones nommées – CSS

```css
#global {
    display:grid;
    grid-template-columns: 1fr 3fr 1fr;
    grid-template-rows: 100px auto 100px;
    grid-template-areas:
        "ent ent ent"
        "men con blt"
        "pdp pdp pdp";
}

header {
    background-color: aqua;
    grid-area: ent;    /* entete */
}

nav {
    background-color: aquamarine;
    grid-area: men;    /* menu */
}

main {
    background-color: coral;
    grid-area: con;    /* contenu */
}

aside {
    background-color: lightgray;
    grid-area: blt;    /* barre latérale */
}

footer {
    background-color: black;
    color:antiquewhite;
    grid-area: pdp;    /* pied de page */
}
```

Zones nommées – Firefox

Zones nommées – validité

- La valeur utilisée pour **grid-template-areas** doit obligatoirement décrire une **grille complète**, sinon elle est considérée comme invalide et la propriété est ignorée.
- Il faut le **même nombre de cellules pour chaque ligne** (si une cellule est vide, on l'indiquera avec un point au lieu d’un nom de zone)
- Si des zones ne sont **pas rectangulaires**, cela sera également considéré comme **invalide**

Zones nommées – usages

- Avec des **media queries**, on peut **adapter un site pour différentes résolutions** en redéfinissant la position des objets sur la grille ou la grille elle-même, ou les deux simultanément.
- On peut également modifier des éléments de **l'interface utilisateur** à l'intérieur d'un site, avec cette technique. Elle n'est pas réservée qu'à la mise en page générale.
  Par exemple, la disposition d'un article de blog