

CSS Grid

**WEB
COURSE
ORT DNIPRO**

ORTDNIPRO.ORG/WEB

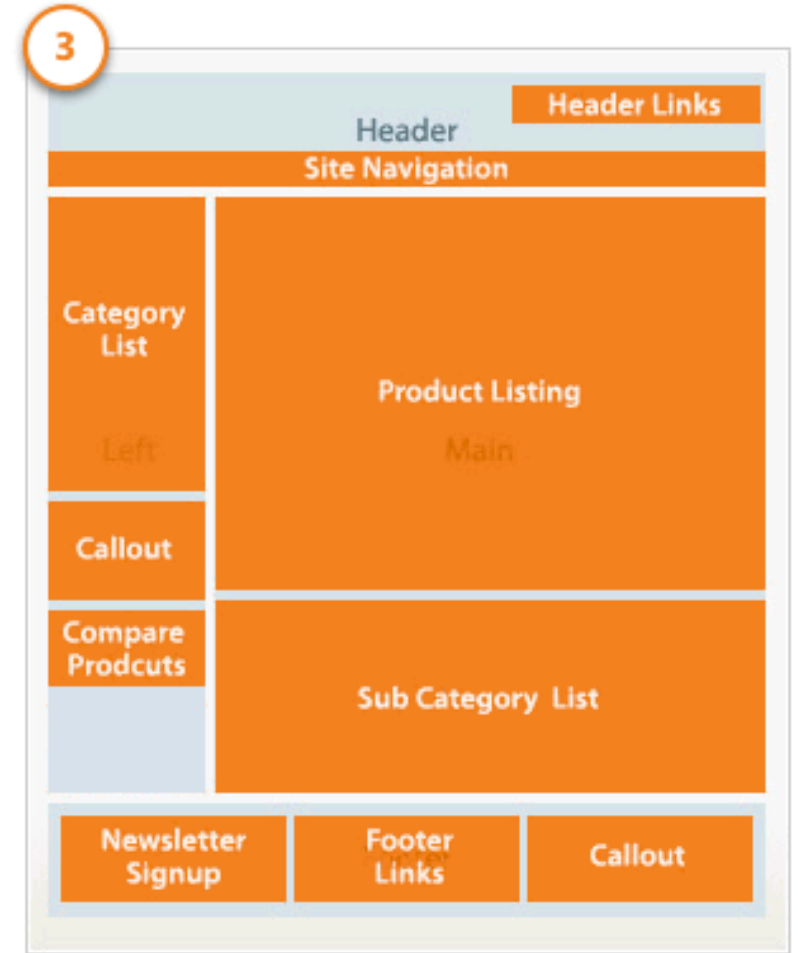
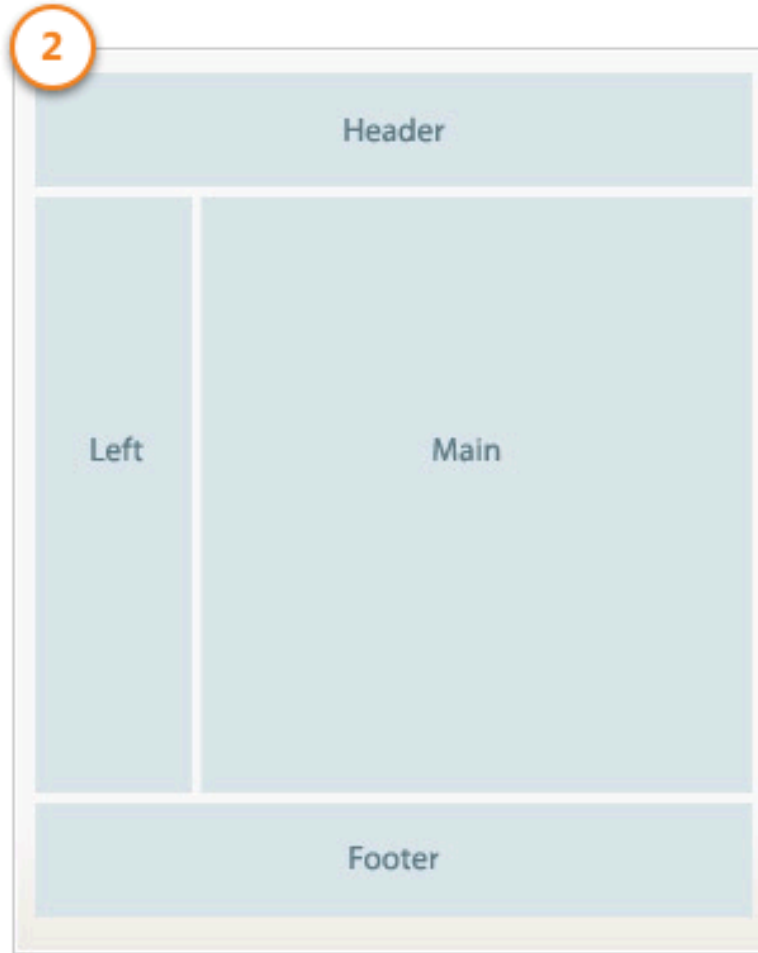
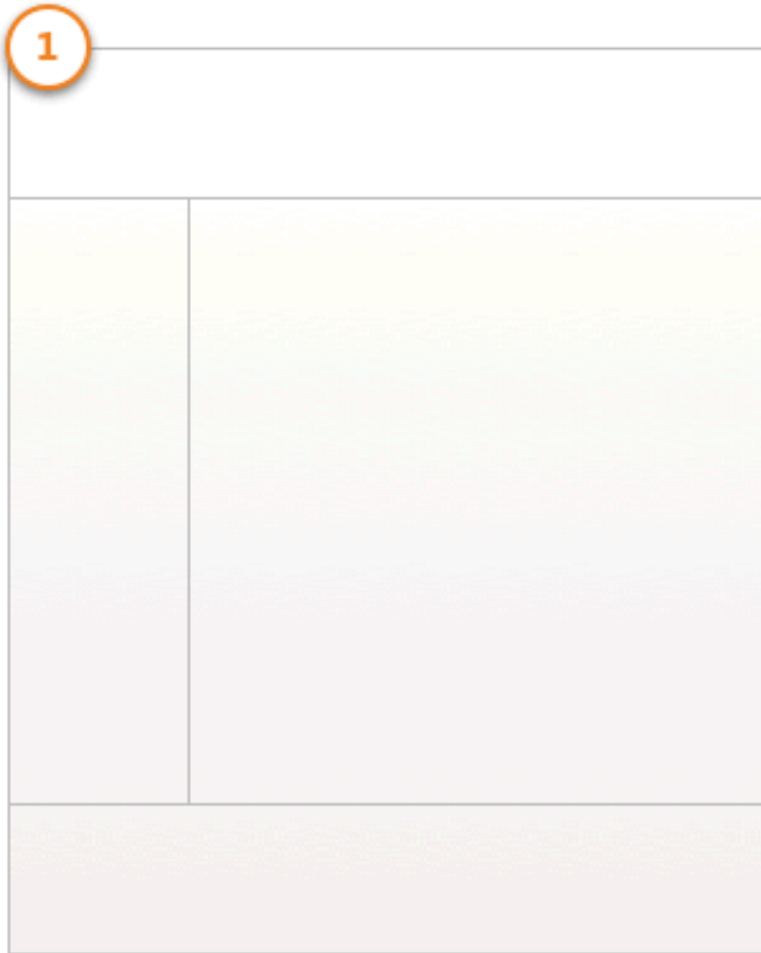
**Сложность размещения
элементов на странице**



Размещение элементов

Размещение элементов на странице, в первую очередь горизонтально (но также и вертикально), всегда было нетривиальной задачей, т.к. до недавнего времени не было инструмента заточенного для решения этой проблем...

Традиционное размещение элементов страницы



Flexbox – не решение
проблемы

У Flexbox есть проблемы со «сложным» расположением компонентов



Sebastien Saunier

CTO at Le Wagon

Get in touch

3,536

contributions

758

followers

119

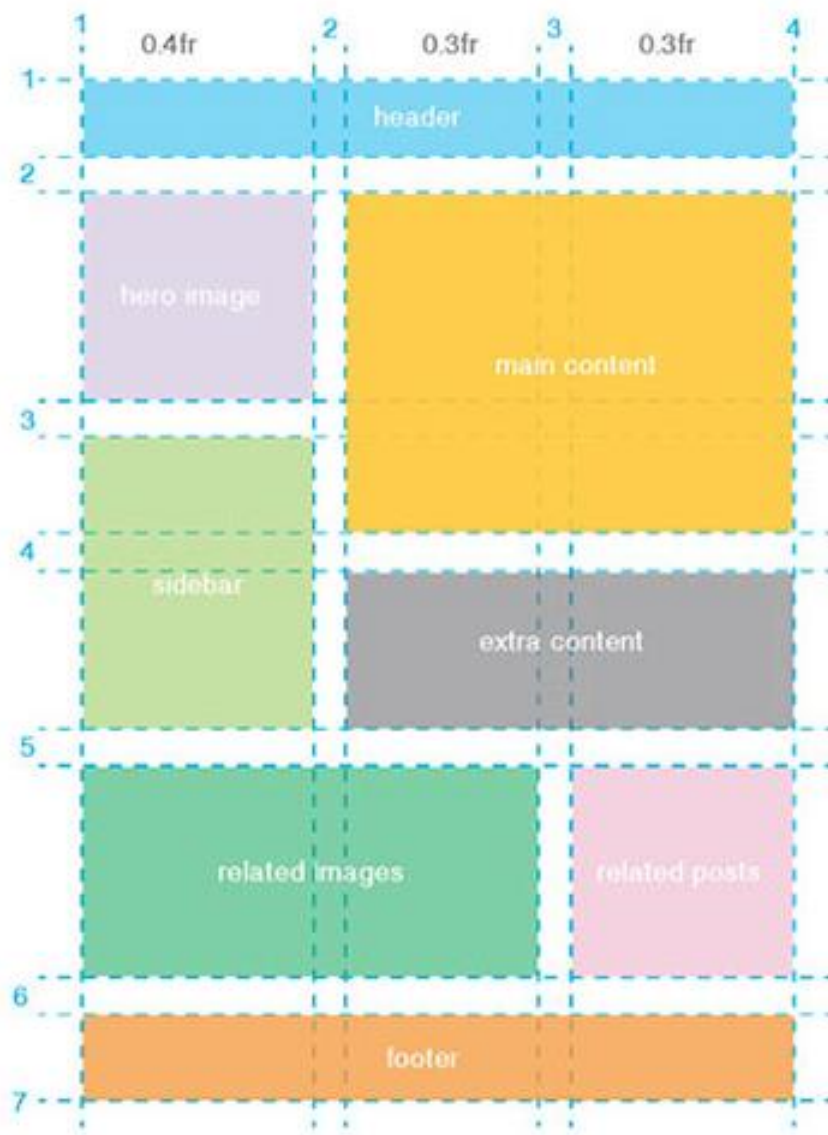
repositories

More about Sebastien

I am a full-stack developer with 9 years of experience working in startups. Scratching your own itch? Working on a startup project? Lots of ideas but frustrated that you can't code them? Confused with agencies' or freelances' quotes? No clue when a developer is talking? Time to learn to code with Le Wagon with a 9-week intensive and immersive bootcamp!

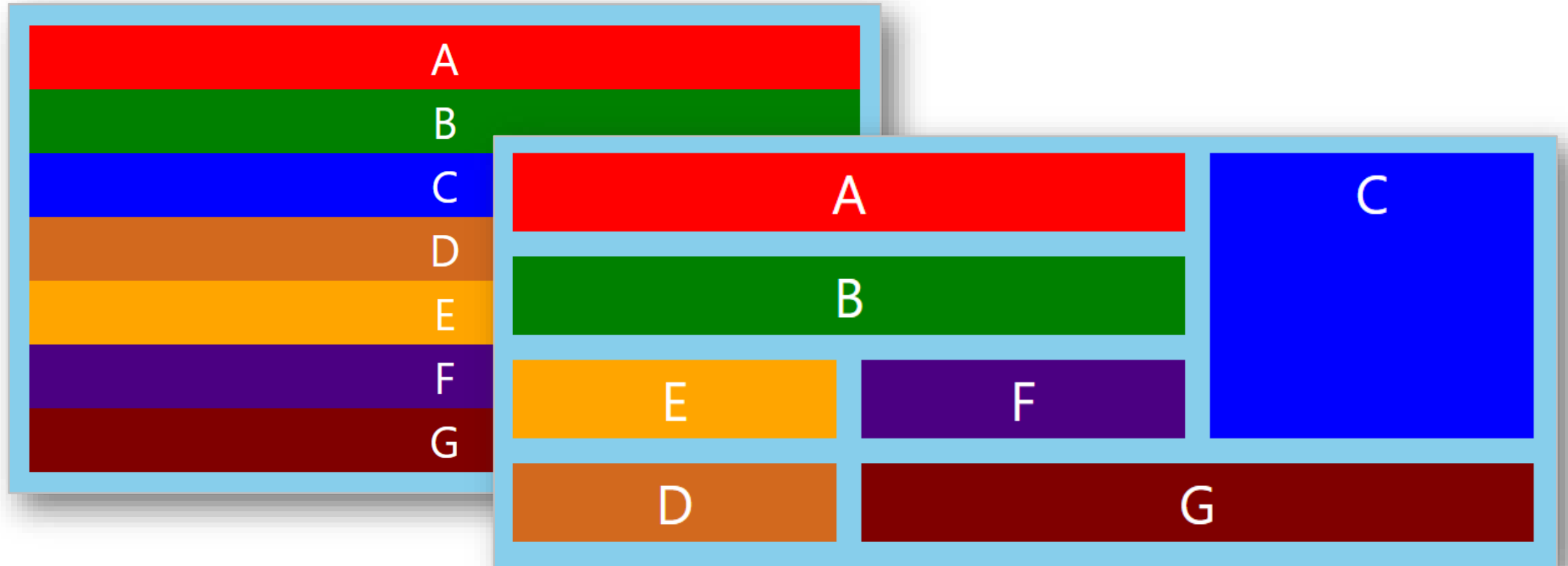
CSS **Grid** — конструктор двухмерных макетов

CSS Grid, всё новое – хорошо забытое старое



CSS Grid – последний, на сегодня, и самый совершенный способ размещения элементов на странице. В отличие от всех ранее перечисленных способов позволяет управлять размещением элементов одновременно и по горизонтали и по вертикали.

CSS Grid

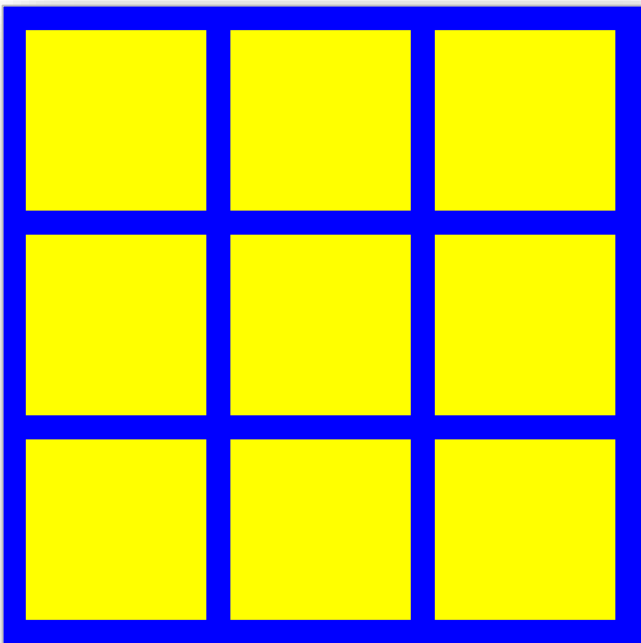


Клонируйте репозиторий этого занятия, в нём будет разметка...

```

1 <main>
2   <div></div>
3   <div></div>
4   <div></div>
5   <div></div>
6   <div></div>
7   <div></div>
8   <div></div>
9   <div></div>
10  <div></div>
11 </main>
12 <style>
13   div {
14     background: yellow;
15   }
16
17   main {
18     background: blue;
19
20     display: grid;
21     grid-template-columns: 150px 150px 150px;
22     grid-template-rows: 150px 150px 150px;
23     grid-gap: 20px;
24     padding: 20px;
25   }
26 </style>

```



CSS Grid

Grid мы можем включить для любого тега, применив **display:grid**, а далее при помощи свойств **grid-template-columns** и **grid-template-rows** задать соответственно: количество и ширину столбцов и строк которые будут в создаваемой сетке. Все дочерние элементы автоматически расположатся в ячейках сетки. Для задания размера столбцов и строк помимо уже известных единиц измерений можно использовать **fr** (например: **2.5fr**) эта единица задаёт часть от свободного пространства. Отступ между ячейками можно задать при помощи свойства **grid-gap**.

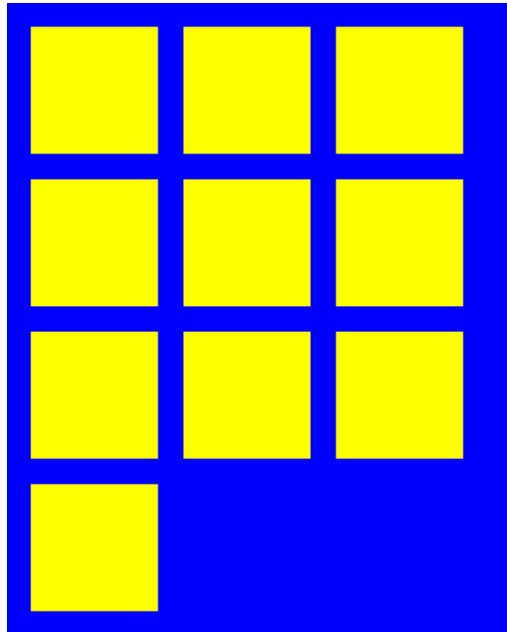
<https://xsltdev.ru/css/grid-template-columns/>

<https://xsltdev.ru/css/grid-template-rows/>

<https://xsltdev.ru/css/grid-gap/>

CSS Grid

```
main {  
  background: blue;  
  
  display: grid;  
  grid-template-columns: repeat(3, 100px);  
  grid-auto-rows: 100px;  
  grid-gap: 20px;  
  padding: 20px;  
}
```



Если мы заранее не знаем сколько у нас будет элементов (и соответственно строк), мы можем задавать высоту для всех возможных строк одним свойством: **grid-auto-rows: 200px;**

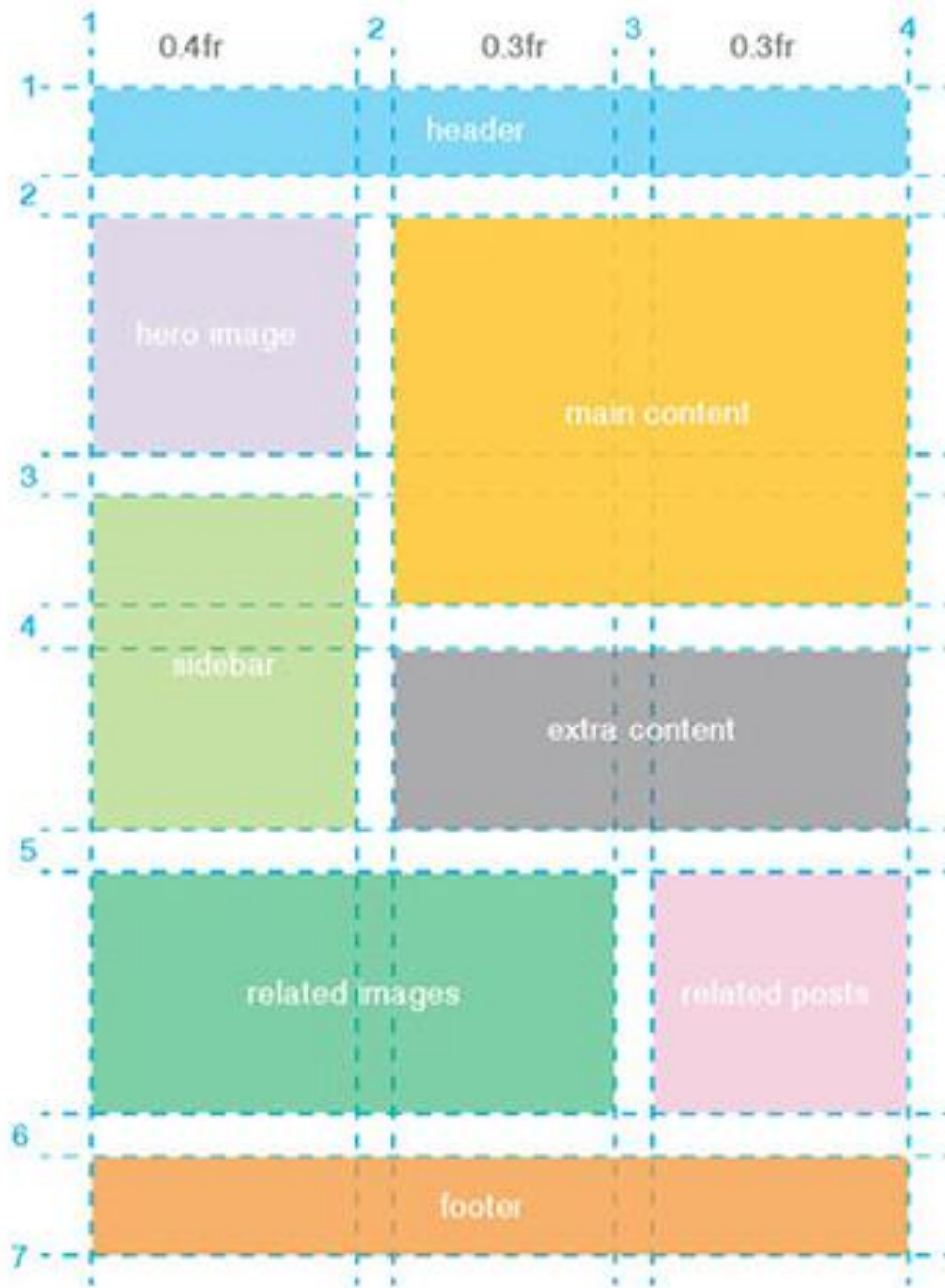
<https://xsltdev.ru/css/grid-auto-rows/>

CSS Grid – minmax()

Функция **minmax()** применяется для задания размеров строк и/или столбцов, и позволяет указать границы между которыми браузер может выбирать размер (браузер будет стремиться задать значение max, но будет учитывать имеющиеся ограничения сетки в целом).

<https://css-live.ru/articles/kak-rabotaet-funkciya-minmax.html>

Grid Lines



Grid Lines

В Grid'е есть понятие линии (**lines**) - линии которая разделяет строки и/или столбцы в сетке. При помощи этих линий (точнее их номеров), мы можем размещать элементы внутри сетки не по порядку, а в нужных нам ячейках. Для этого нам нужны свойства: **grid-column-start**, **grid-column-end**, **grid-row-start**, **grid-row-end** – которые, соответственно, задают номера начальной и конечной вертикальной линии, и начальной и конечной горизонтальной линии. *Но это далеко не самый удобный способ...*

<https://xsltdev.ru/css/grid-column-start/>

<https://xsltdev.ru/css/grid-column-end/>

<https://xsltdev.ru/css/grid-row-start/>

<https://xsltdev.ru/css/grid-row-end/>

CSS Grid Areas

```

<main>
  <div class="cat"></div>
  <div class="dog"></div>
  <div class="fish"></div>
  <div class="bird"></div>
</main>

```

```

.cat{ background: yellow; grid-area: cat; }
.dog{ background: pink; grid-area: dog; }
.fish{ background: cyan; grid-area: fish; }
.bird{ background: orange; grid-area: bird; }

```

```

main {
  display: grid;
  background: blue;
  grid-gap: 20px;
  padding: 20px;
  grid-template-columns: 1fr 1fr 1fr;
  grid-template-rows: 100px 100px 100px;
  grid-template-areas: "cat    cat    dog"
                      "fish   fish   dog"
                      "bird   .     dog"
}

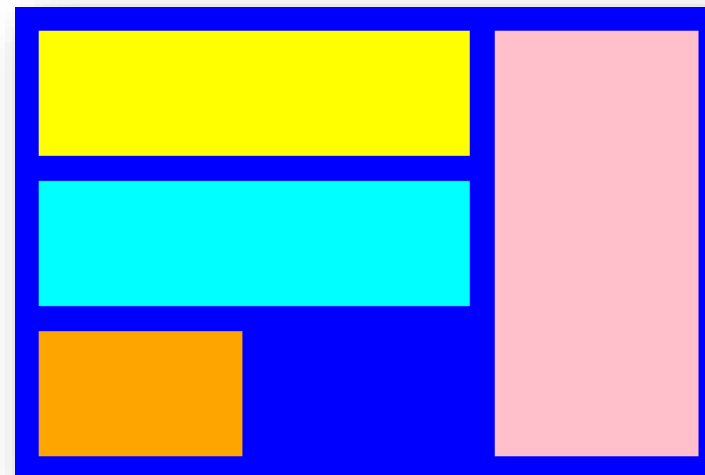
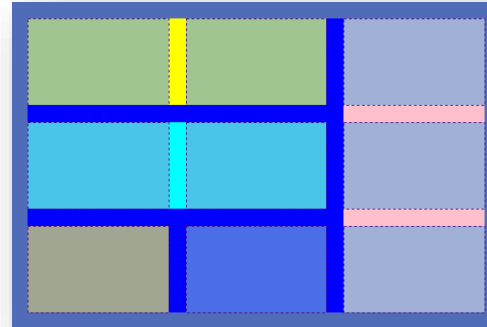
```

Grid Areas

CSS Grid позволяет в текстовом виде формировать макет. Для этого у каждого элемента в сетке должно быть задано имя посредством свойства **grid-area**. А при помощи свойств **grid-template-areas** мы задаём расположение элементов в сетке при помощи заданных имён элементов.

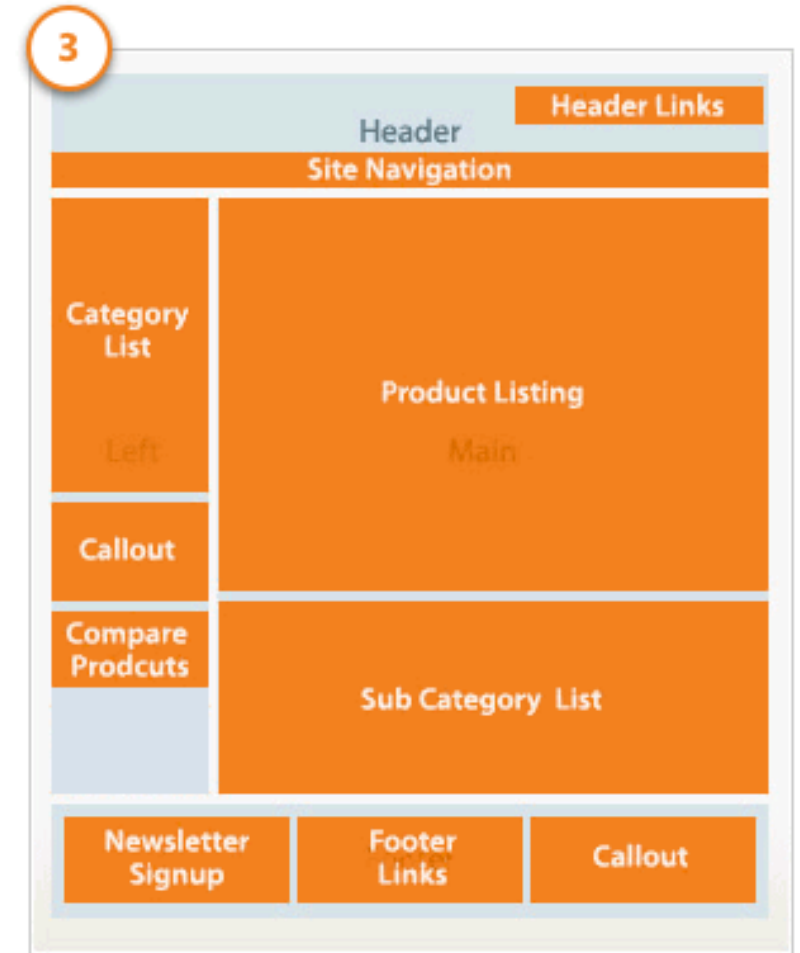
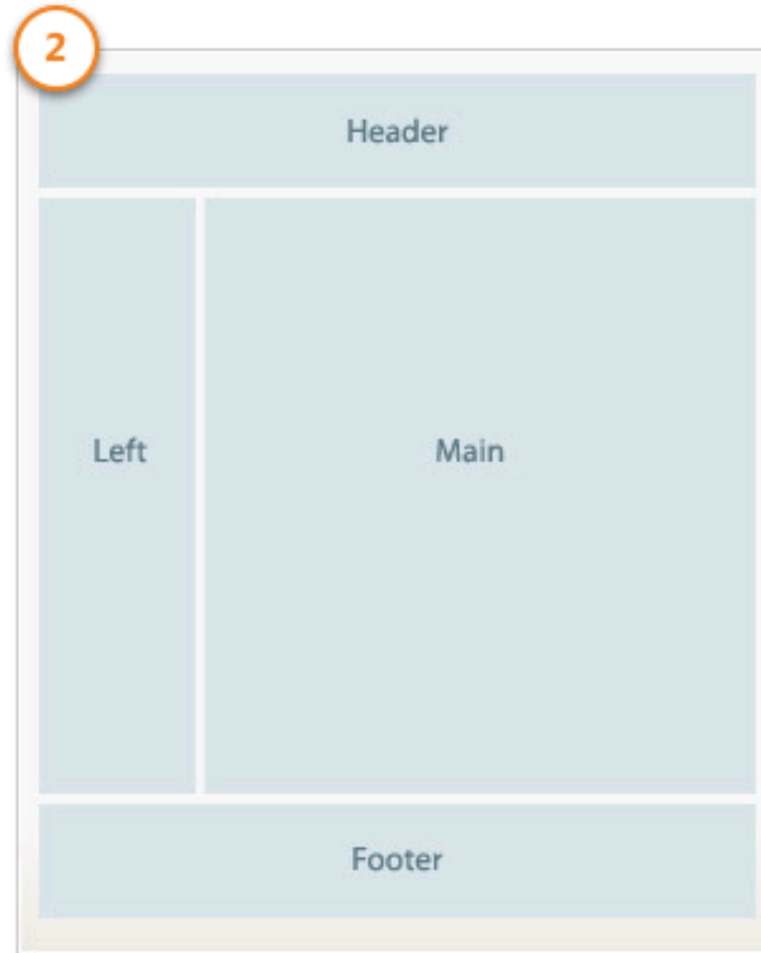
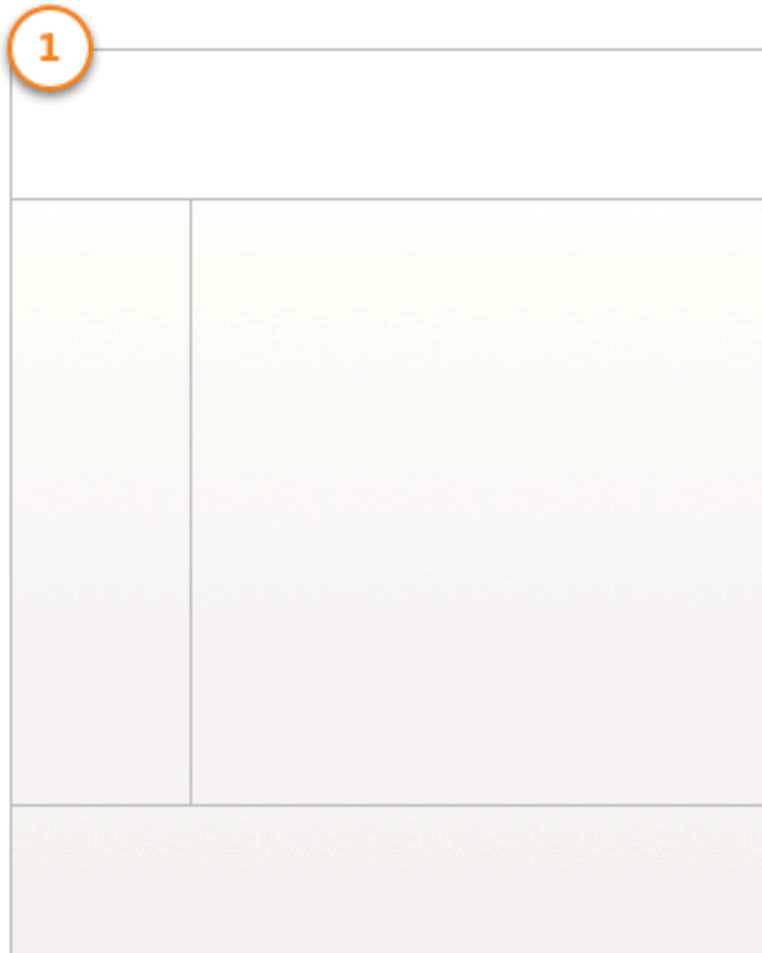
<https://xsltdev.ru/css/grid-area/>

<https://xsltdev.ru/css/grid-template-areas/>



Немного практики

Традиционное размещение элементов страницы



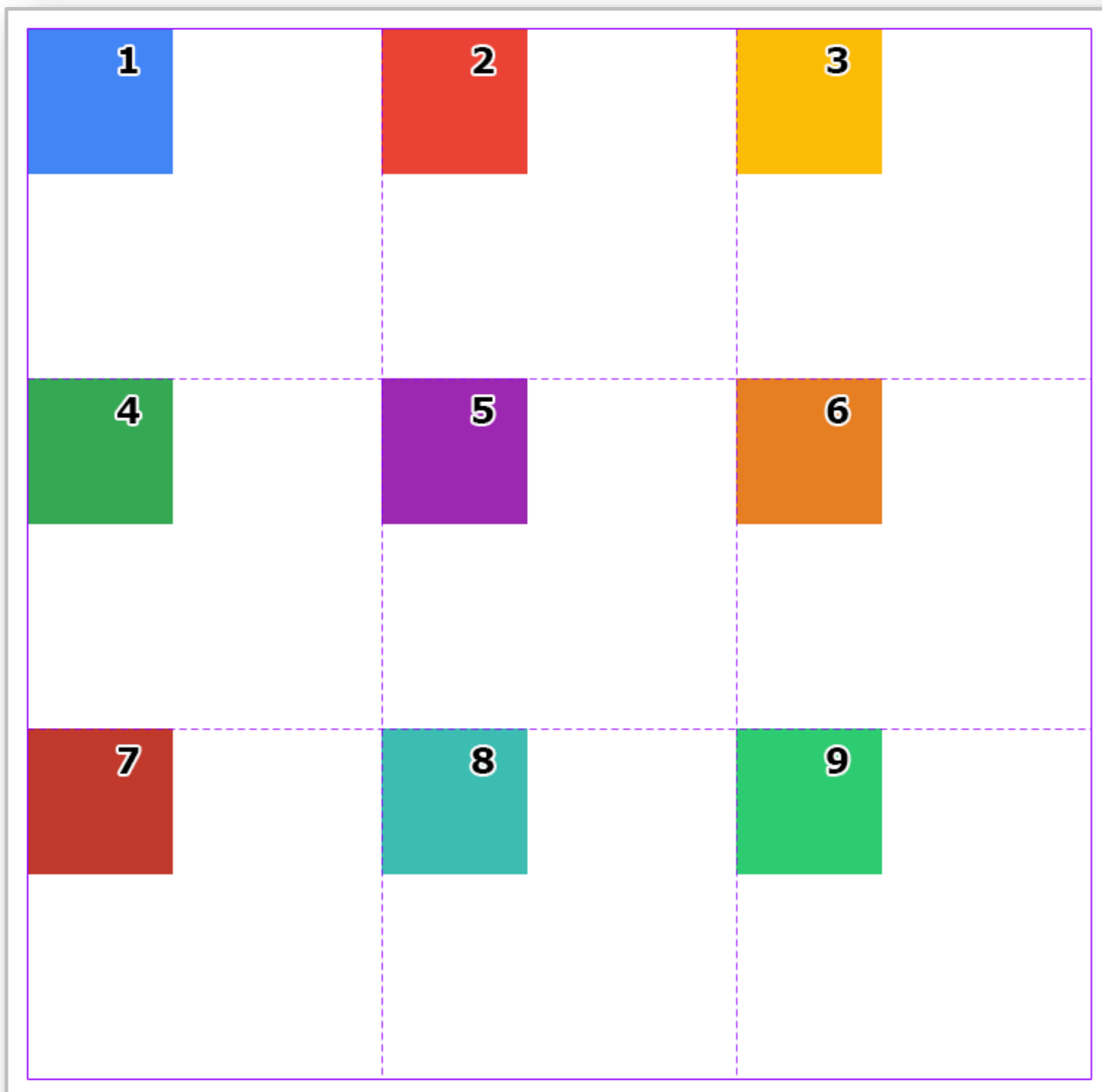
CSS Grid позволяет легко разметить страницу расположив элементы по своим местам.

Будет полезным

1	2	3	4	5	6	7	8
9	10 grid-column: span 3			11	12	13	14
15	16	17	18	19	12 grid-row: span 3	20	21
22	23 grid-column: span 3 grid-row: span 3			24		25	26
27				28		30	31
32				33	34	35	36
37	38	39	40	41	42	43	44

Еще раз о CSS Grid

<https://youtu.be/-fDqBEjfzGo?t=39>



О выравнивании элементов в CSS Grid

CSS Grid даёт возможность выравнивать содержимое ячеек сетки, а также распределять ячейки сетки в пределах родительского элемента.

<http://tpverstak.ru/grid/>

Домашнее задание

Пройти игру по CSS Grid!

GRID GARDEN

Level 25 of 28

Now there is a 75 pixel column of weeds on the left side of your garden. 3/5 of the remaining space is growing carrots, while 2/5 has been overrun with weeds.


Use `grid-template-columns` with a combination of `px` and `fr` units to make the necessary columns.

```
1 #garden {  
2   display: grid;  
3   grid-template-columns: 75px 3fr 2fr  
4   grid-template-rows: 100%;  
5 }  
6  
7  
8  
9  
10  
11  
12  
13  
14
```

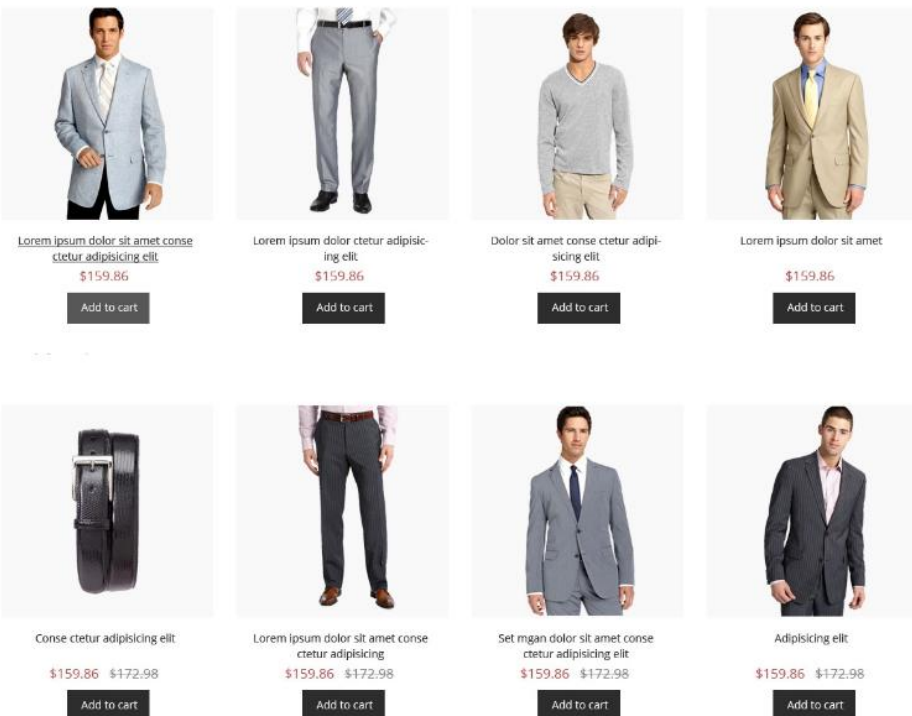
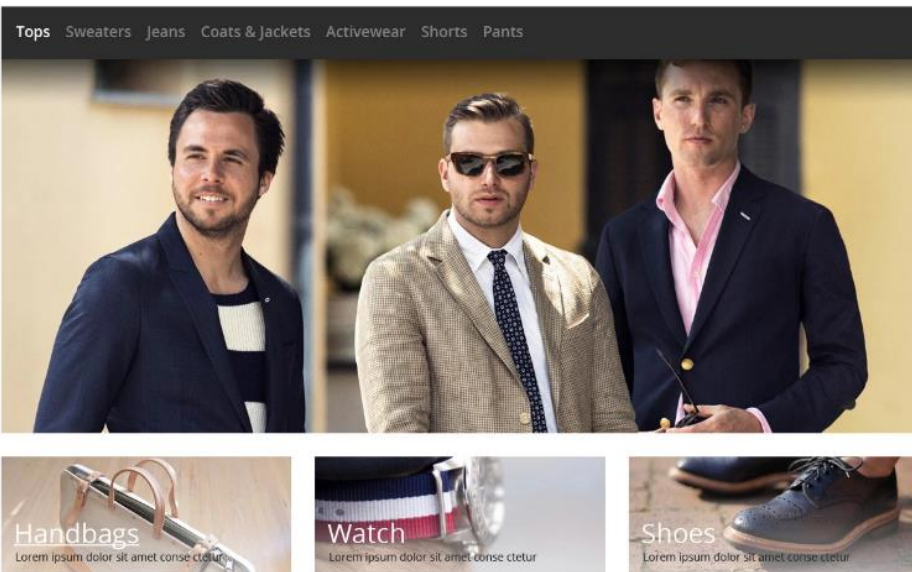
Next

Grid Garden is created by [CodePip](#) • [GitHub](#) • [Twitter](#) • [English](#)

Want to learn CSS flexbox? Play [Flexbox Froggy](#).

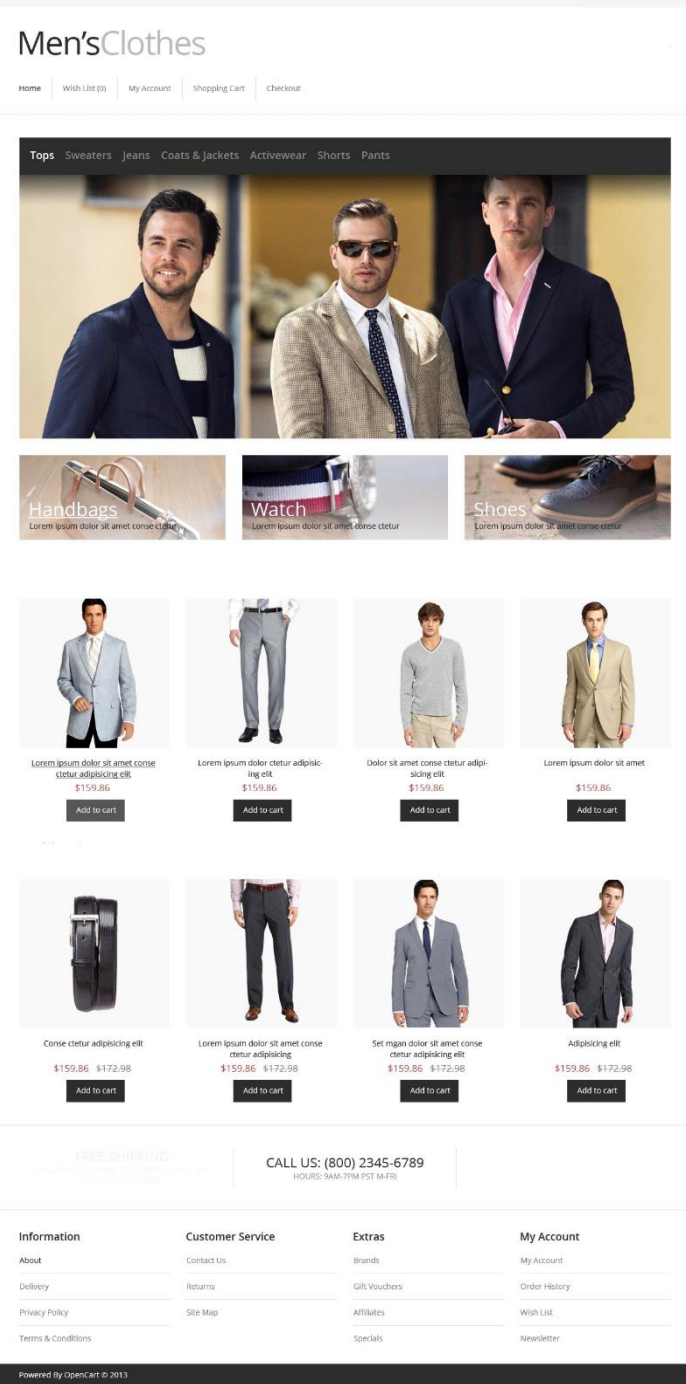


<https://cssgridgarden.com/>



Домашнее задание

CSS Grid и Bootstrap вам
в ПОМОЩЬ...



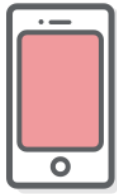
Домашнее задание++

CSS Grid и Bootstrap вам в ПОМОЩЬ...

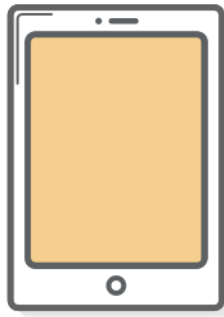
Макет доступен в репозитории занятия, в каталоге `./homework-layout`

**К следующему
занятию...**

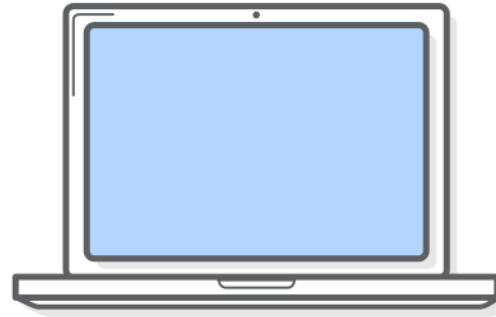
Адаптивная вёрстка



MOBILE



TABLET



DESKTOP

Предварительные знания – лучший помощник в обучении, поэтому к следующему занятию жду, что **посмотрите небольшой ролик о медиа-запросах.**

<https://youtu.be/M-xc1EOMOIE>