

# ОСОБЕННОСТИ ВЕРСТКИ ГРАФИКИ ДЛЯ МОБИЛЬНЫХ УСТРОЙСТВ



СЕМЕН БОЙКО



# СЕМЕН БОЙКО



[simonderus@gmail.com](mailto:simonderus@gmail.com)



[fb.me/simonboycko](https://fb.me/simonboycko)



# ПЛАН ЗАНЯТИЯ

1. [Графика](#)
2. [Иконочный шрифт](#)
3. [SVG](#)
4. [Особенности верстки форм на мобильных устройствах](#)
5. [Отображение телефона синим цветом на iOS](#)
6. [Выравнивание flexbox-элементов](#)



# ГРАФИКА

---

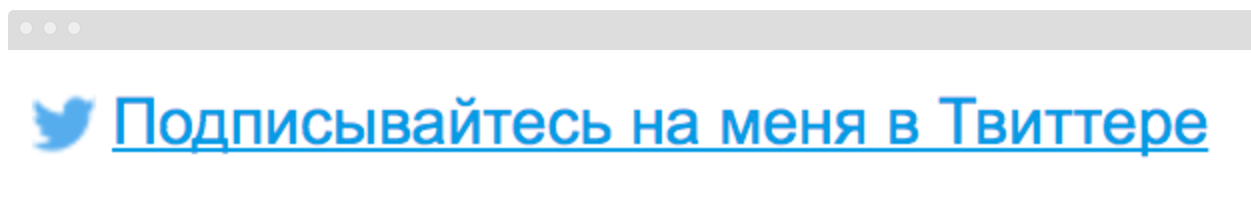
## БЛОК ПОДПИСКИ В ТВИТТЕРЕ

Выполняя задачи по верстке, мы постоянно сталкиваемся с графикой. Но до этого момента мы не рассматривали ее особенности на мобильных устройствах. Для наглядности сверстаем блок «Подписывайтесь на меня в Твиттере».



## ПРОВЕРЯЕМ НА IPHONE

А теперь давайте посмотрим, как выглядит тот же блок на iPhone:

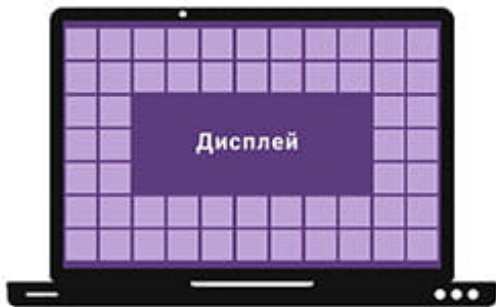


Иконка выглядит нечеткой или, как говорят разработчики, «мыльной».



*Физический пиксель — это лампочка, которая горит определенным цветом.*

*Любой экран состоит из пикселей.*



# СТИЛИ ДЛЯ ВЫВОДА КАРТИНКИ

Для вывода изображения написан следующий CSS:

```
1  .follow-me__social {  
2      background-image: url("tw_16x16.png");  
3      background-repeat: no-repeat;  
4      background-position: left center;  
5      background-size: 16px;  
6  }
```

С помощью свойства `background-size` указываем браузеру, чтобы он зарезервировал место под изображение размером 16 физических пикселей по горизонтали и столько же по вертикали.

Далее браузер пытается загрузить изображение, указанное в свойстве `background-image`, и вставить его.



# АНАЛИЗ ИЗОБРАЖЕНИЯ

Но перед тем, как загрузить изображение, браузер анализирует его размеры. И здесь возможны три случая:

1. Размер изображения соответствует размеру, указанному в `background-size`. Браузер вставляет картинку такой, какая она есть.
2. Размер изображения больше указанного в `background-size` размера. Браузер уменьшает изображение до указанных размеров.
3. Размер изображения меньше указанного в `background-size` размера. Браузер растягивает картинку до указанных размеров. При этом страдает качество растровой графики.

# ПОЯВЛЕНИЕ RETINA

По такому принципу изображения на сайте отображались до 2007 года.

В 2007 году компания Apple разработала новый вид экранов с повышенной четкостью (Retina-экраны). У таких экранов увеличилось количество пикселей на единицу площади.



# ФОРМУЛА РАСЧЕТА ПИКСЕЛЕЙ

Браузеры уже не могли рассчитывать размеры области под изображения старым способом. И появилась следующая формула:

$$P_{px} = CSS_{px} * DPR$$

Где:

- $P_{px}$  — физический пиксель на экране устройства;
- $CSS_{px}$  — виртуальный пиксель, указанный в CSS;
- DPR (Device Pixel Ratio) — коэффициент соотношения физического пикселя к виртуальному.

[Список устройств с характеристиками](#)

# СКОЛЬКО ПИКСЕЛЕЙ НА КАРТИНКУ?

Теперь посчитаем сколько физических пикселей потребовалось для отображения иконка на iPhone:

$$16\text{px} * 2 = 32\text{px};$$

Но иконка имеет размер 16x16px. Соответственно произошла ситуация, когда размер изображения меньше указанного в свойстве `background-size`. Поэтому браузеру пришлось растянуть картинку, и она потеряла свое качество.

## БОЛЬШЕ РАЗМЕР ИКОНКИ

Один из способов решения: для всех экранов загружать иконки размером 32x32px, а в CSS указать размер следующим образом:

```
1  .follow-me__social {  
2      background-image: url("tw_32x32.png");  
3      background-repeat: no-repeat;  
4      background-position: left center;  
5      background-size: 16px;  
6  }
```

В таком случае, если DPR равен 1, то браузер уменьшит ее до 16x16. А если DPR равен 2, то он отобразит в исходном размере.



# ИКОНОЧНЫЙ ШРИФТ



*Иконочный шрифт – это шрифт, в котором символами являются иконки.*

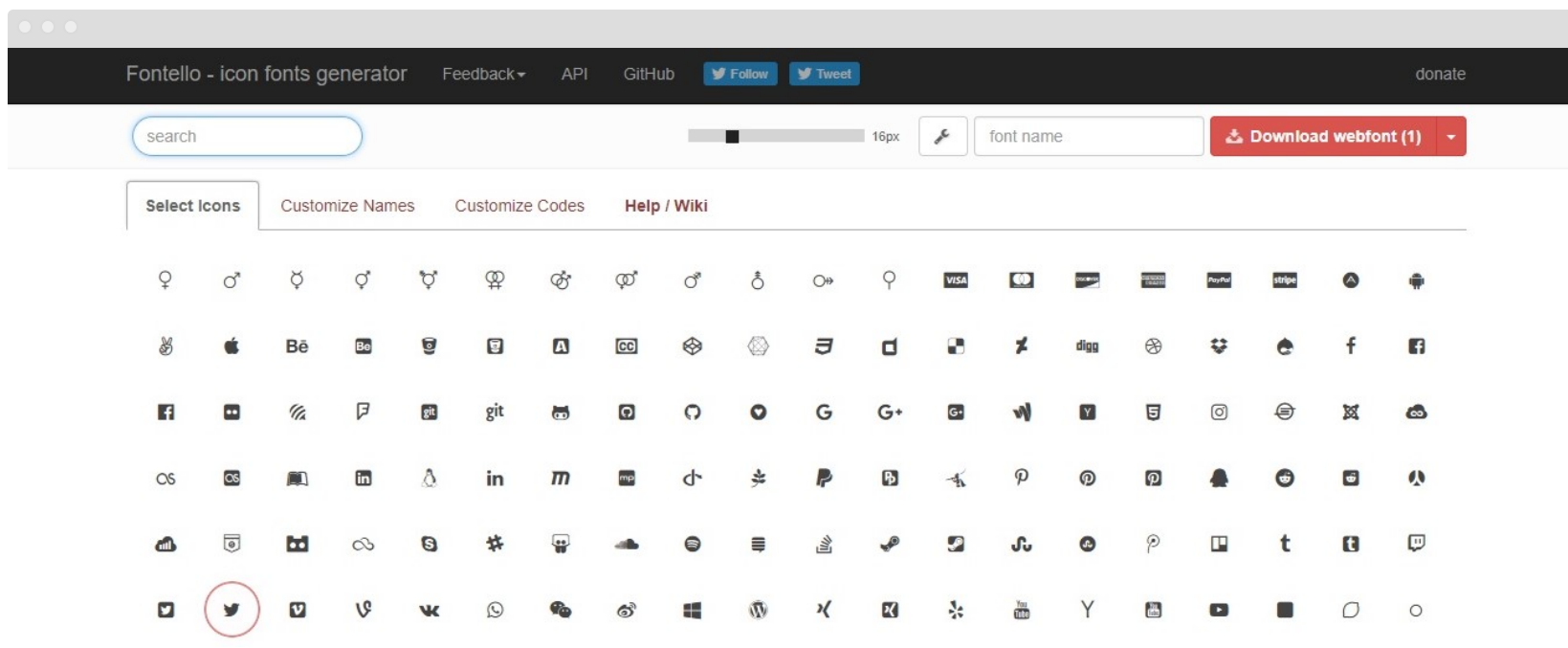
*Для верстальщика это означает, что он может работать с графикой, как с текстом. Например, применять свойства `font-size`, `color` и прочие.*



# НАЧАЛО РАБОТЫ

Для начала нужно открыть сайт, откуда мы возьмем иконочный шрифт. Существует множество таких сайтов, с помощью которых можно создать свой набор. Например, <http://fontello.com/>.



Выберем нужные иконки из шрифта Font Awesome и нажмем кнопку `Download webfont` в правом верхнем углу.





## ПАПКА СО ШРИФТАМИ

После разархивации архива появится папка font, в которой находятся файлы шрифта:

 fontello.eot	01.09.2017 18:00	Файл "EOT"	6 КБ
 fontello	01.09.2017 18:00	Файл "SVG"	1 КБ
 fontello	01.09.2017 18:00	Файл шрифта Tru...	5 КБ
 fontello.woff	01.09.2017 18:00	Файл "WOFF"	3 КБ
 fontello.woff2	01.09.2017 18:00	Файл "WOFF2"	3 КБ

Скопируем их и добавим в папку с проектом.

# ПОДКЛЮЧАЕМ ШРИФТЫ

Теперь в CSS-файле нужно подключить их как шрифт, используя директиву `@font-face`:

```
1  @font-face {  
2      font-family: fontello;  
3      src: url("font/fontello.woff2?42978351") format("woff2"),  
4           url("font/fontello.woff?42978351") format("woff");  
5      font-weight: normal;  
6      font-style: normal;  
7  }  
8  
9  .follow-me__social {  
10     background-image: url("tw_32x32.png");  
11     background-repeat: no-repeat;  
12     background-position: left center;  
13     background-size: 16px;  
14 }
```

## УДАЛИМ СВОЙСТВА ФОНА

Теперь изменим свойства для каскада `.follow-me__social`. Удалим свойства `background-image`, `background-repeat`, `background-position`, `background-size`.

## ДОБАВИМ ПСЕВДОЭЛЕМЕНТ

Для того, чтобы использовать иконочные шрифты, нам понадобятся псевдоэлементы. В свойстве `content` укажем код иконки. Также добавим `font-family`, `font-size` и `color` для иконки:

```
1  @font-face {
2      font-family: fontello;
3      src: url("font/fontello.woff2?42978351") format("woff2"),
4           url("font/fontello.woff?42978351") format("woff");
5      font-weight: normal;
6      font-style: normal;
7  }
8
9  .follow-me__social::before {
10     content: "\f099";
11     font-family: fontello;
12     font-size: 16px;
13     color: #76a9ea;
14 }
```

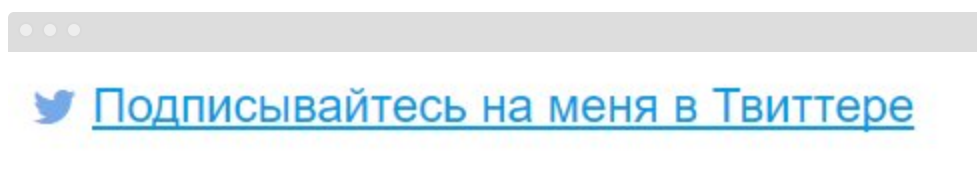
## СПОЗИЦИОНИРУЕМ ИКОНКУ

Используем абсолютное позиционирование, чтобы расположить иконку согласно дизайну:

```
1  .follow-me__social{
2      position: relative;
3  }
4
5  .follow-me__social::before{
6      content: "\f099";
7      font-family: fontello;
8      font-size: 16px;
9      color: #76a9ea;
10     position: absolute;
11     top: 5px;
12     left: 0;
13 }
```

## ПРОВЕРЯЕМ РЕЗУЛЬТАТ

Если откроем браузер, то внешне получим тот же результат. Но на Retina-дисплее иконка не будет «мылиться»:



# РАЗНЫЕ ИКОНКИ ДЛЯ РАЗНЫХ УСТРОЙСТВ

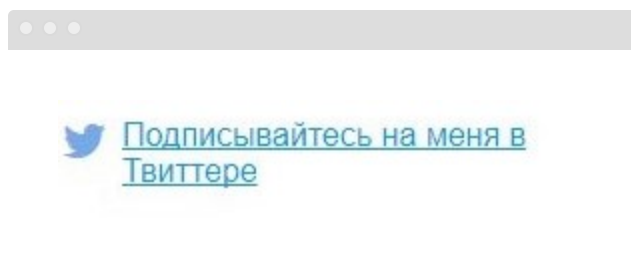
Используя медиавыражения, мы сделаем для широкоформатных экранов иконку размером 16x16px, а для телефонов — 24x24px.

```
1  @media (min-width: 481px) {
2    .follow-me__social {
3      padding: 5px 20px;
4    }
5
6    .follow-me__social::before {
7      font-size: 16px;
8    }
9  }
10
11 @media (max-width: 480px) {
12   .follow-me__social {
13     padding: 5px 20px 5px 30px;
14   }
15
16   .follow-me__social::before{
17     font-size: 24px;
18   }
19 }
```

---

# БОЛЬШАЯ ИКОНКА НА МОБИЛЬНОМ

На мобильном иконка больше и выглядит отлично:



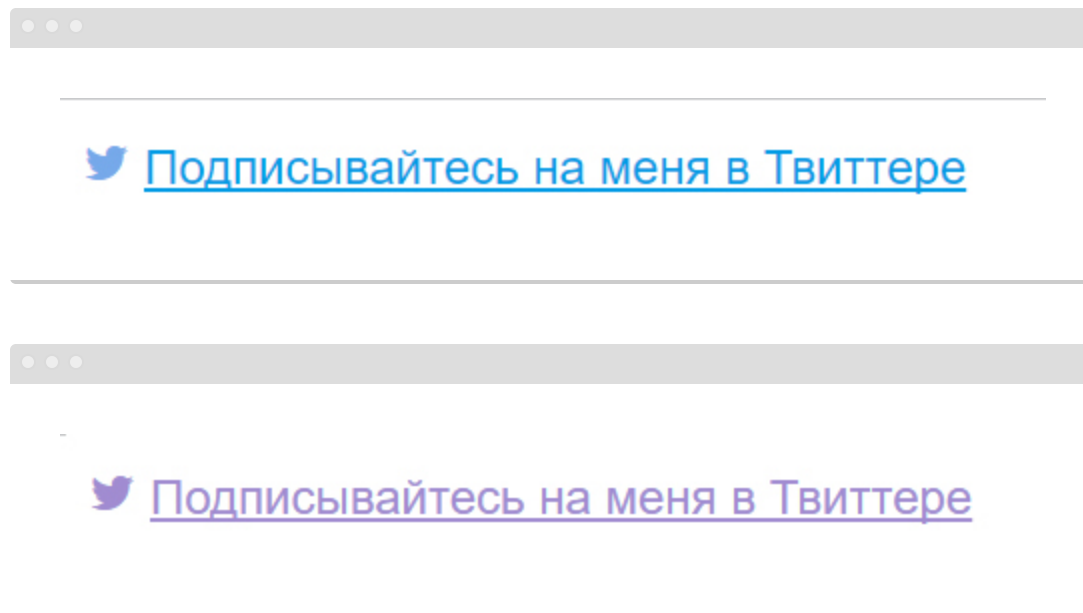


# HOVER-ЭФФЕКТ

С использованием иконочных шрифтов очень легко можно менять цвет иконок. Допустим поступила задача сделать для больших экранов hover-эффект к блоку.

```
1  @media (min-width: 481px) {  
2      .follow-me__social {  
3          padding: 5px 20px;  
4      }  
5  
6      .follow-me__social:hover,  
7      .follow-me__social:hover::before {  
8          color: #a18cd1;  
9      }  
10  
11     .follow-me__social::before {  
12         font-size: 16px;  
13     }  
14 }
```

# ИКОНКА МЕНЯЕТ СВОЙ ЦВЕТ



# НЕДОСТАТКИ ИКОНОЧНЫХ ШРИФТОВ

Между тем, использование иконочных шрифтов имеет один недостаток. При таком подходе наши иконки являются текстом, а это совсем странно и несемантично.

На самом деле, иконочные шрифты были популярны довольно давно. И пик их популярности пришелся на то время, когда в большинстве браузеров не поддерживалось SVG.



**SVG**



*SVG – это масштабируемая векторная графика, созданная Консорциумом Всемирной паутины (W3C). Данный тип графики имеет все преимущества иконочных шрифтов, но при этом является графикой.*



РАСТР  
.jpeg .gif .png



ВЕКТОР  
.svg

# ЗАГРУЗКА ВНЕШНИХ ФАЙЛОВ

Первый способ использования SVG: сохранить SVG-иконку отдельным файлом и подключить ее с помощью свойства `background`.

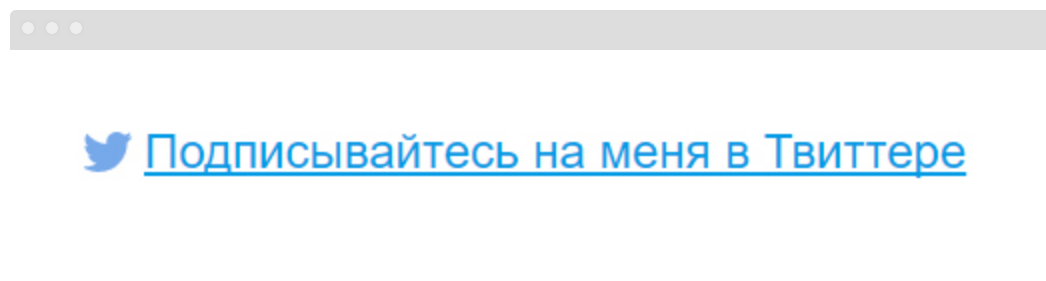
Вернемся к примеру, где мы сделали разный размер иконок для различных платформ. Подключим файл с помощью свойства `background-image`, а размер зададим, используя `background-size`.

## ЗАМЕНИМ ИКОНКИ НА SVG

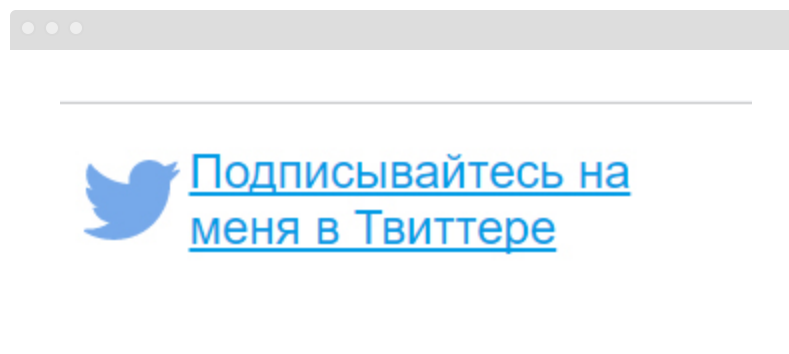
```
1  .follow-me__social {
2      background-image: url("twitter.svg");
3      background-repeat: no-repeat;
4      background-position: left center;
5  }
6
7  @media (min-width: 481px) {
8      .follow-me__social {
9          padding: 5px 20px;
10         background-size: 16px;
11     }
12 }
13
14 @media (max-width: 480px) {
15     .follow-me__social {
16         padding: 5px 20px 5px 35px;
17         background-size: 32px;
18     }
19 }
```

# ПРОВЕРЯЕМ НА УСТРОЙСТВАХ

Десктоп:



Мобильное устройство:







## НЕДОСТАТОК ВНЕШНЕГО ФАЙЛА

Мы не можем изменять цвет иконки. При использовании данной техники, иконка становится как будто за стеклом и до нее не достучаться.

# ВСТРОЕННЫЙ SVG В HTML

Мы можем использовать SVG прямо в HTML. Для примера вставим иконку Твиттера.

```
1 <div class="follow-me">
2   <a href="#0" class="follow-me__social">
3     <svg class="follow-me__icon" viewBox="0 0 26 28">
4       <path fill="#039be5" d="M25.312 6.375c-0.688 1-1.547 1.891-2.531 2.609 0.016 0.219
5         0.016 0.438 0.016 0.656 0 6.672-5.078 14.359-14.359 14.359-2.859
6         0-5.516-0.828-7.75-2.266 0.406 0.047 0.797 0.063 1.219 0.063 2.359 0 4.531-0.797
7         6.266-2.156-2.219-0.047-4.078-1.5-4.719-3.5 0.313 0.047 0.625 0.078 0.953 0.078 0.453
8         0 0.906-0.063 1.328-0.172-2.312-0.469-4.047-2.5-4.047-4.953v-0.063c0.672 0.375 1.453
9         0.609 2.281 0.641-1.359-0.906-2.25-2.453-2.25-4.203 0-0.938 0.25-1.797 0.688-2.547
10        2.484 3.062 6.219 5.063 10.406 5.281-0.078-0.375-0.125-0.766-0.125-1.156 0-2.781
11        2.25-5.047 5.047-5.047 1.453 0 2.766 0.609 3.687 1.594 1.141-0.219 2.234-0.641
12        3.203-1.219-0.375 1.172-1.172 2.156-2.219 2.781 1.016-0.109 2-0.391 2.906-0.781z">
13     </path>
14   </svg>
15   <span class="follow-me__social-label">Подписывайтесь на меня в Твиттере</span>
16 </a>
17 </div>
18 </div>
```

---

# ОГРОМНАЯ ИКОНКА



[Подписывайтесь на меня в Твиттере](#)

---

## ЗАДАЕМ РАЗМЕРЫ

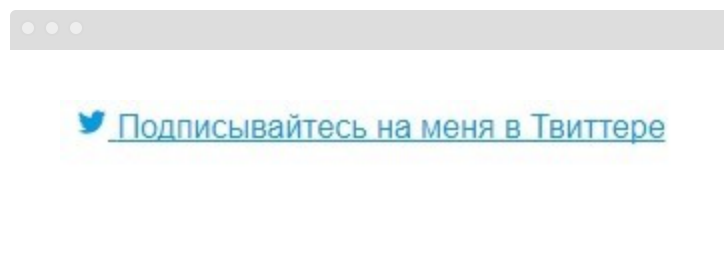
Такое отображение нас не устраивает, и мы можем это изменить, используя CSS. Зададим для иконки размеры:

```
1  .follow-me__social {
2      display: inline-block;
3      font-family: Arial, sans-serif;
4      color: #039be5;
5  }
6
7  .follow-me__icon {
8      width: 16px;
9      height: 16px;
10 }
```

---

# ПОДХОДЯЩИЙ РАЗМЕР

Теперь на мониторах у нас блок выглядит как надо:



# МЕНЯЕМ РАЗМЕР ДЛЯ МОБИЛЬНЫХ

Изменим CSS так, чтобы было два размера иконок:

```
1  .follow-me__social {
2      display: inline-block;
3      font-family: Arial, sans-serif;
4      color: #039be5;
5  }
6
7  @media (min-width: 481px) {
8      .follow-me__icon {
9          width: 16px;
10         height: 16px;
11     }
12 }
13
14 @media (max-width: 480px) {
15     .follow-me__icon {
16         width: 24px;
17         height: 24px;
18     }
19 }
```

[Live Demo](#)

# fill

Атрибут `fill` используют для заливки иконки.

Попробуем изменить цвет иконки на `#a18cd1`.

```

1 <div class="follow-me">
2   <a href="#0" class="follow-me__social">
3     <svg class="follow-me__icon" viewBox="0 0 26 28">
4       <path fill="#a18cd1" d="M25.312 6.375c-0.688 1-1.547 1.891-2.531 2.609 0.016 0.219
5         0.016 0.438 0.016 0.656 0 6.672-5.078 14.359-14.359 14.359-2.859
6         0-5.516-0.828-7.75-2.266 0.406 0.047 0.797 0.063 1.219 0.063 2.359 0 4.531-0.797
7         6.266-2.156-2.219-0.047-4.078-1.5-4.719-3.5 0.313 0.047 0.625 0.078 0.953 0.078 0.453
8         0 0.906-0.063 1.328-0.172-2.312-0.469-4.047-2.5-4.047-4.953v-0.063c0.672 0.375 1.453
9         0.609 2.281 0.641-1.359-0.906-2.25-2.453-2.25-4.203 0-0.938 0.25-1.797 0.688-2.547
10        2.484 3.062 6.219 5.063 10.406 5.281-0.078-0.375-0.125-0.766-0.125-1.156 0-2.781
11        2.25-5.047 5.047-5.047 1.453 0 2.766 0.609 3.687 1.594 1.141-0.219 2.234-0.641
12        3.203-1.219-0.375 1.172-1.172 2.156-2.219 2.781 1.016-0.109 2-0.391 2.906-0.781z">
13     </path>
14   </svg>
15   <span class="follow-me__social-label">Подписывайтесь на меня в Твиттере</span>
16 </a>
17 </div>

```

[Live Demo](#)

---

# ЛИШЕНИЕ ГИБКОСТИ

Но использование инлайн-атрибута `fill` для задания цвета иконки лишает SVG графику огромной гибкости.

Чтобы сделать решение универсальным, можно использовать SVG в связке с CSS.



---

# ИСПОЛЬЗУЕМ CSS

В качестве примера сделаем изменение цвета текста и иконки при наведении мыши.

Для начала добавим состояние `hover`:

```
1 | .follow-me__social:hover {  
2 |     color: #a18cd1;  
3 | }
```

## МЕНЯЕМ `fill` ЧЕРЕЗ CSS

Теперь самое интересное. В CSS существует аналогичное свойство `fill`, которое так же, как и атрибут, изменяет заливку SVG. Попробуем задать цвет иконки, используя его:

```
1 | .follow-me__icon {  
2 |     fill: #039be5;  
3 | }
```

А при наведении на элемент `.follow-me__social` изменим его цвет:

```
1 | .follow-me__social:hover .follow-me__icon{  
2 |     fill: #a18cd1;  
3 | }
```

# УДАЛЯЕМ `fill` ИЗ РАЗМЕТКИ

Но сейчас ничего не работает! Потому что в теге `<svg>` все еще находится атрибут `fill`. Его нужно удалить:

```

1 <div class="follow-me">
2   <a href="#0" class="follow-me__social">
3     <svg class="follow-me__icon" viewBox="0 0 26 28">
4       <path d="M25.312 6.375c-0.688 1-1.547 1.891-2.531 2.609 0.016 0.219
5         0.016 0.438 0.016 0.656 0 6.672-5.078 14.359-14.359 14.359-2.859
6         0-5.516-0.828-7.75-2.266 0.406 0.047 0.797 0.063 1.219 0.063 2.359 0 4.531-0.797
7         6.266-2.156-2.219-0.047-4.078-1.5-4.719-3.5 0.313 0.047 0.625 0.078 0.953 0.078 0.453
8         0 0.906-0.063 1.328-0.172-2.312-0.469-4.047-2.5-4.047-4.953v-0.063c0.672 0.375 1.453
9         0.609 2.281 0.641-1.359-0.906-2.25-2.453-2.25-4.203 0-0.938 0.25-1.797 0.688-2.547
10        2.484 3.062 6.219 5.063 10.406 5.281-0.078-0.375-0.125-0.766-0.125-1.156 0-2.781
11        2.25-5.047 5.047-5.047 1.453 0 2.766 0.609 3.687 1.594 1.141-0.219 2.234-0.641
12        3.203-1.219-0.375 1.172-1.172 2.156-2.219 2.781 1.016-0.109 2-0.391 2.906-0.781z">
13     </path>
14   </svg>
15   <span class="follow-me__social-label">Подписывайтесь на меня в Твиттере</span>
16 </a>
17 </div>

```

[Live Demo](#)

# ВСЕ РАБОТАЕТ!



 [Подписывайтесь на меня в Твиттере](#)



 [Подписывайтесь на меня в Твиттере](#)

# ГДЕ БРАТЬ SVG?


Чтобы ответить на этот вопрос, откроем файл tw.svg в любом текстовом редакторе. Увидим следующий код:

tw.svg

```
1 <svg xmlns="http://www.w3.org/2000/svg" viewBox="0 0 26 28">
2   <path fill="#039be5" d="M25.312 6.375c-0.688 1-1.547 1.891-2.531 2.609 0.016 0.219 0.016 0.438 0.016 0.656 0 6.672-5.078 14.359-14.359 14.359-2.859 0-5.516-0.828-7.7
*   6.219 5.063 10.406 5.281-0.078-0.375-0.125-0.766-0.125-1.156 0-2.781 2.25-5.047 5.047-5.047 1.453 0 2.766 0.609 3.687 1.594 1.141-0.219 2.234-0.641 3.203-1.219-0.375
3   </svg>
-
```

Копируем код и вставляем в HTML.

Во время приемки макета у дизайнера нужно проверить, приложил ли дизайнер SVG иконки к макету, и если нет — запросить их.



# ОСОБЕННОСТИ ВЕРСТКИ ФОРМ НА МОБИЛЬНЫХ УСТРОЙСТВАХ

# ФОРМЫ НА МОБИЛЬНЫХ

Давайте рассмотрим, как элементы формы отображаются на мобильных устройствах. Для этого мы сверстали форму отправки сообщения.

**Введите ваше имя**

Например, Иван

**Введите ваш E-mail**

Например, test@ya.ru

**Выберите тему сообщения**

Предложение по работе

**Напишите сообщение**

Привет...

Отправить

# ПРОВЕРИМ НА IPHONE

**Введите ваше имя**

Например, Иван

**Введите ваш E-mail**

Например, test@ya.ru

**Выберите тему сообщения**

Предложение по работе ▼

**Напишите сообщение**

Привет...

Отправить



# ПЕРЕБИВАЕМ СТИЛИ БРАУЗЕРА

В браузере есть CSS-правила, которые определены в операционной системе. В нашем случае в iOS есть свои стили для `input` и `select`.

Для начала уберем скругление у полей с помощью `border-radius`:

```
1  .field {  
2    border: 2px solid #000000;  
3    box-sizing: border-box;  
4    padding: 10px;  
5    width: 100%;  
6    font-family: inherit;  
7    font-size: 100%;  
8    border-radius: 0;  
9  }
```

# appearance

С помощью свойства `appearance` можно сделать любой элемент выглядящим как элемент формы со стандартными стилями операционной системы. А нам, наоборот, нужно убрать их. И для этого случая у свойства существует значение `none`, которое мы и будем использовать.

# ВЕНДОРНЫЕ ПРЕФИКСЫ

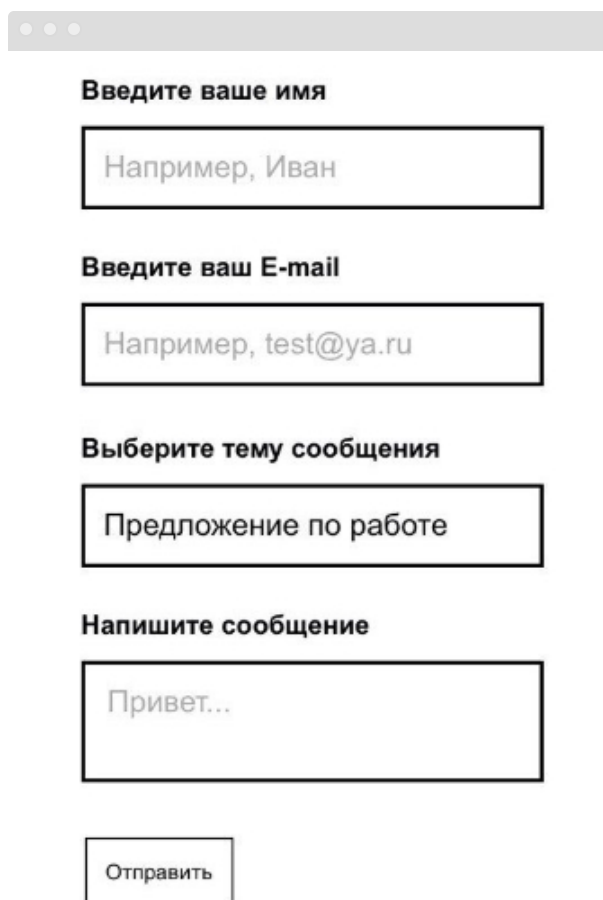
Но само свойство `appearance` [не поддерживается браузерами](#).

Поэтому следует использовать вендорные префиксы `-webkit` и `-moz`.

```
1  .field {  
2    border: 2px solid #000000;  
3    box-sizing: border-box;  
4    padding: 10px;  
5    width: 100%;  
6    font-family: inherit;  
7    font-size: 100%;  
8    border-radius: 0;  
9    -webkit-appearance: none;  
10   -moz-appearance: none;  
11 }
```

# select СЛОМАЛСЯ

Тень у полей пропала, и `select` стал отображаться почти как надо. Но у него пропала «стрелочка».



The image shows a web browser window with a form. The form has four input fields and a submit button. The first field is for a name, the second for an email, the third for a message topic, and the fourth for the message body. The third field, which should be a dropdown menu, is currently displaying a single text option instead of a list of options with arrows. The submit button is labeled 'Отправить'.

Введите ваше имя

Например, Иван

Введите ваш E-mail

Например, test@ya.ru

Выберите тему сообщения

Предложение по работе

Напишите сообщение

Привет...

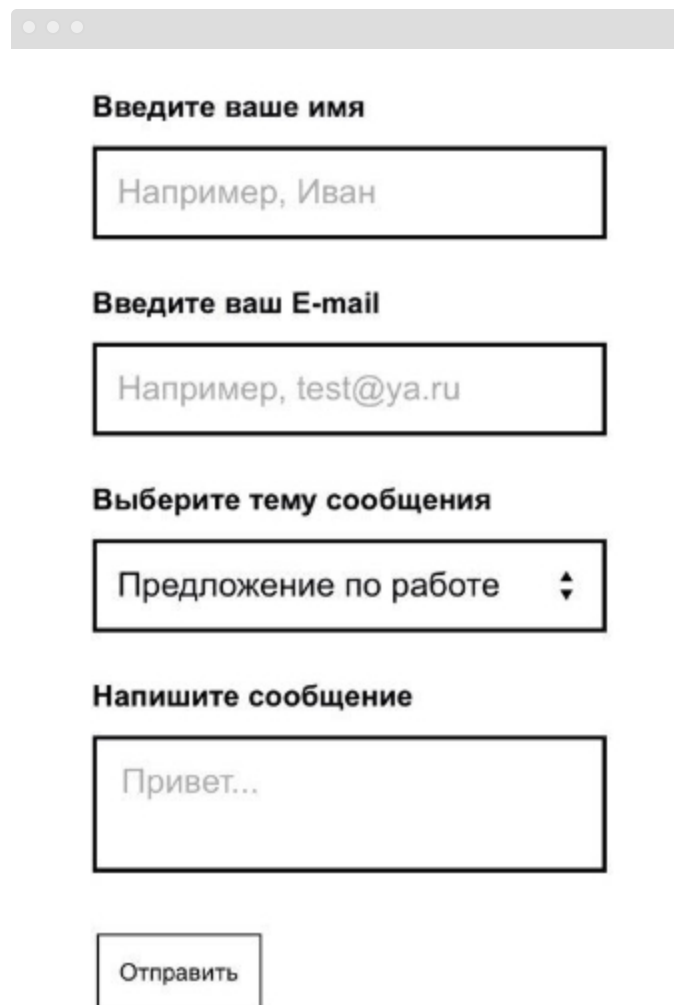
Отправить

## ЗАМЕНЯЕМ ИКОНКУ

Поэтому добавим свою иконку, которая заменит стандартную.

```
1  select {  
2      background-image: url("select.svg");  
3      background-repeat: no-repeat;  
4      background-position: 98% center;  
5      padding-right: 35px;  
6  }
```

# УЗНАВАЕМЫЙ `select`



Введите ваше имя

Введите ваш E-mail

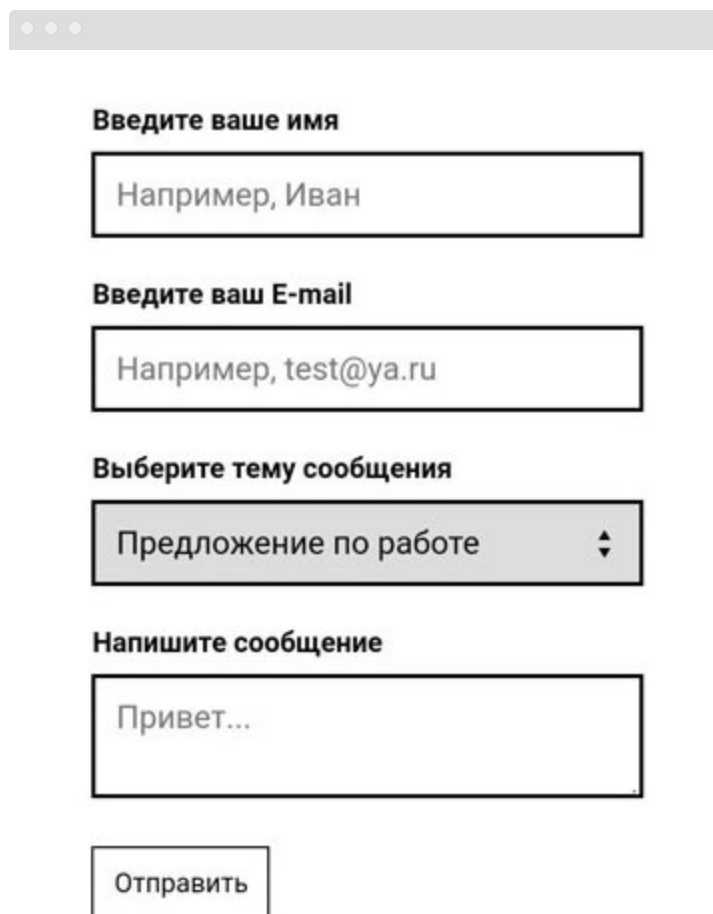
Выберите тему сообщения

Напишите сообщение

Отправить

# ПРОВЕРЯЕМ НА ANDROID

Как видим, у выпадающего списка появился серый фон.



Введите ваше имя

Введите ваш E-mail

Выберите тему сообщения

Предложение по работе

⬆

Напишите сообщение

Отправить

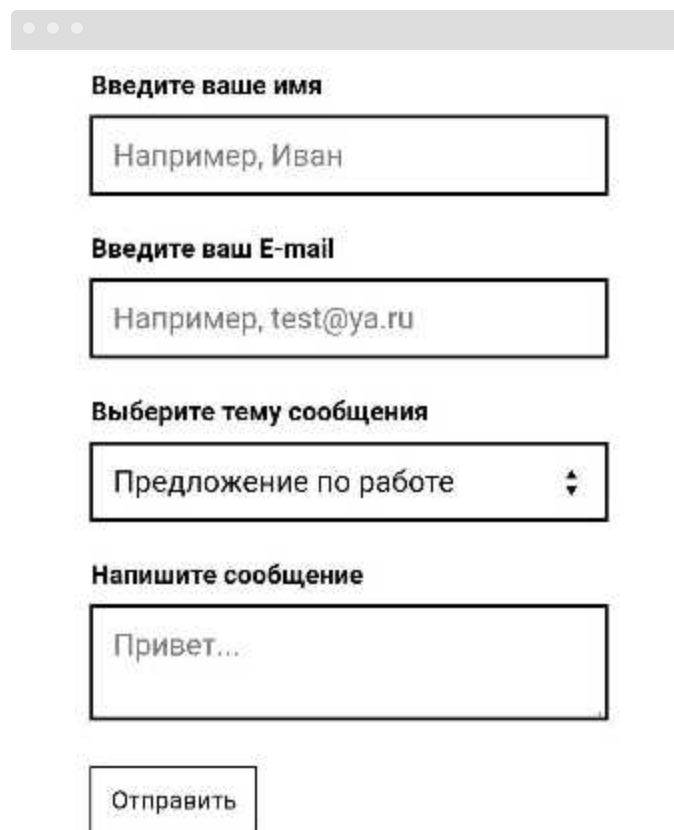
## МЕНЯЕМ ФОН

Для того, чтобы исправить это, добавим `background-color` со значением `#ffffff`.

```
1  select {  
2      background-image: url("select.svg");  
3      background-repeat: no-repeat;  
4      background-position: 98% center;  
5      background-color: #ffffff;  
6      padding-right: 35px;  
7  }
```



# ОТЛИЧНО НА ВСЕХ ПЛАТФОРМАХ



Введите ваше имя

Введите ваш E-mail

Выберите тему сообщения

Напишите сообщение

Отправить

[Live Demo](#)

---

# ОТОБРАЖЕНИЕ ТЕЛЕФОНА СИНИМ ЦВЕТОМ НА IOS

## БЛОК КОНТАКТОВ

```
1 <div class="contacts">
2   <h3 class="contacts__title">Наши контакты</h3>
3   <div class="contacts__row">
4     <span class="contacts__item">+7-000-000-00-00</span>
5   </div>
6   <div class="contacts__row">
7     <span class="contacts__item">+7-000-111-00-00</span>
8   </div>
9 </div>
```

# ОТКРОЕМ НА IPHONE

A screenshot of a mobile application window. At the top, there is a title bar with three small white circles on a gray background. Below the title bar, the text "Наши контакты" is displayed in a large, bold, black font. Underneath this title, there are two lines of text, each preceded by a small blue icon of a telephone handset. The first line is "+7-000-000-00-00" and the second line is "+7-000-111-00-00". Both lines of text are blue and underlined. A thin gray horizontal line is positioned below the second line of text.

## Наши контакты

[+7-000-000-00-00](tel:+7-000-000-00-00)

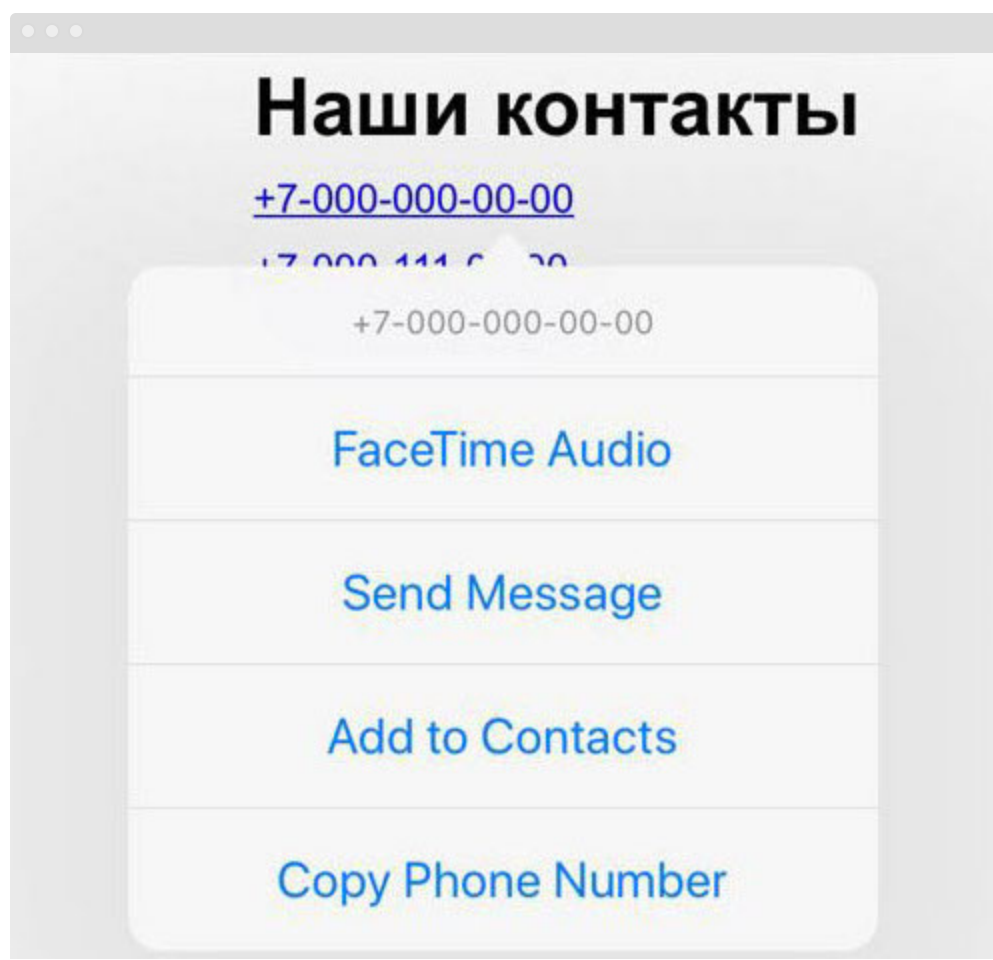
[+7-000-111-00-00](tel:+7-000-111-00-00)

### [Live Demo](#)

Как видим, ссылки приобрели синий цвет и подчеркивания. Это произошло потому, что номер телефона стал ссылкой с протоколом `tel:`. Скорее всего это случилось, потому что разработчики iOS решили позаботиться о пользователях.

# ЗВОНОК ПО КЛИКУ

Теперь при клике по номеру откроется окно с предложением позвонить по нему:



## МЕНЯЕМ СТИЛИ

Но мы не можем оставить такой цвет, потому что по дизайн-макету цвет телефона должен совпадать с цветом текста и не иметь подчеркивания.

Поэтому добавим стили для блока `.contacts__item`:

```
1 | .contacts__item {  
2 |     color: inherit;  
3 |     text-decoration: none;  
4 | }
```

# СНОВА ПРОВЕРЯЕМ НА IPHONE

## Наши контакты

[+7-000-000-00-00](tel:+7-000-000-00-00)

[+7-000-111-00-00](tel:+7-000-111-00-00)

# СТИЛИ БРАУЗЕРА ПРИОРИТЕТНЕЕ

К сожалению, ничего не изменилось. Стили мобильного браузера оказываются приоритетнее наших. Попробуем обратиться к блоку по селектору тега и атрибута:

```
1  a[href^=tel] {  
2      color: inherit;  
3      text-decoration: none;  
4  }
```

[Live Demo](#)





# НУЖНЫЙ РЕЗУЛЬТАТ



## Наши контакты

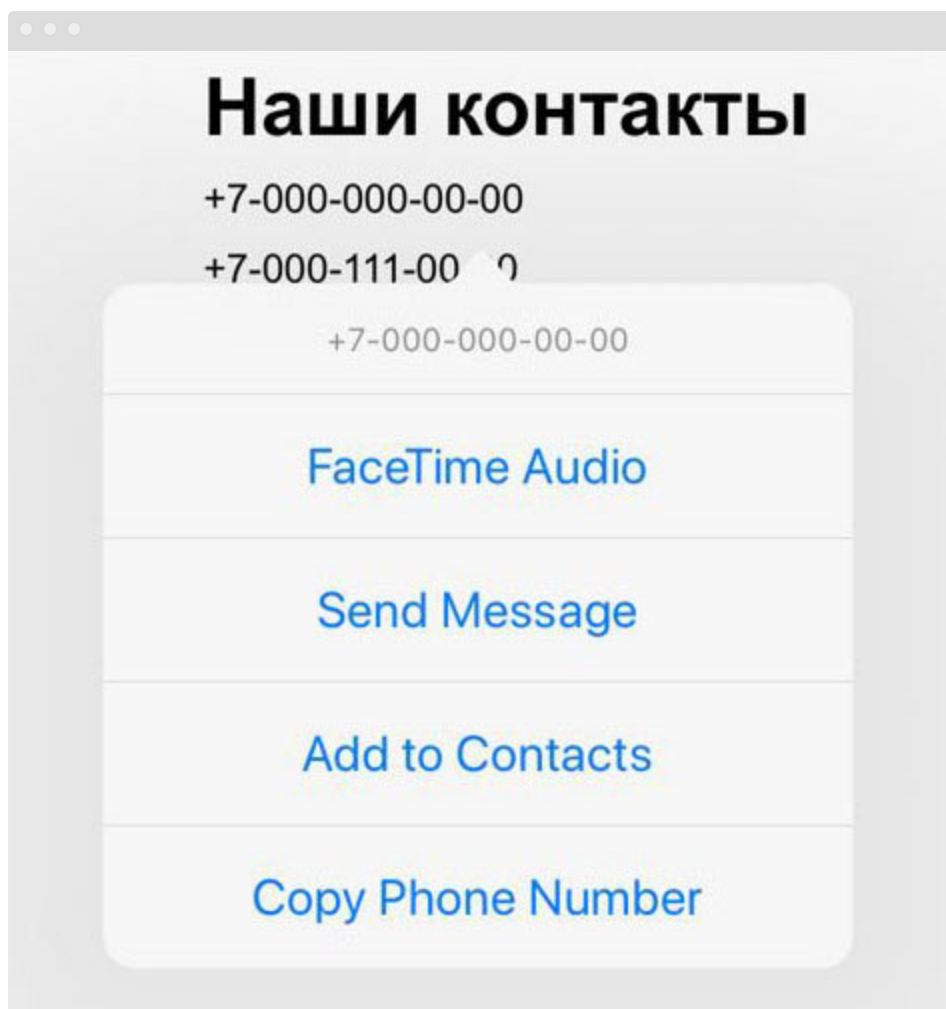
+7-000-000-00-00

+7-000-111-00-00

---

# КЛИК ПО СПРЯТАННЫМ ТЕЛЕФОНАМ

Но попробуем кликнуть на блок:



# ЗАПРЕЩАЕМ РАСПОЗНАВАТЬ ТЕЛЕФОНЫ

Видим, что несмотря на то, что внешний вид блоков с телефонами изменился, функционал, встроенный разработчиками мобильного браузера, остался.

Но, если хотите отменить это поведение и запретить мобильному браузеру распознавать блоки с номерами телефонов, можно добавить специальный метатег в секцию `head`:

```
<meta name="format-detection" content="telephone=no">
```

[Live Demo](#)

# ОТМЕНЕН КЛИК

## Наши контакты

+7-000-000-00-00

+7-000-111-00-00

Но так лучше не делать. Пользователям гораздо удобнее просто нажать на номер телефона, чем выделять его и копировать.



# **ВЫРАВНИВАНИЕ FLEXBOX-ЭЛЕМЕНТОВ**

# БЛОК С НОВОСТЬЮ

Сверстаем блок с новостью в блоге:

20  
JUN

## The Basics of Binary and Hex, Explained in Video Form

Lorem ipsum dolor sit amet, consectetur adipisicing elit. Deleniti cum error deserunt! Nostrum, architecto! Temporibus ullam inventore totam ea necessitatibus repellat placeat ad saepe laboriosam natus error vitae numquam unde eligendi cum maiores rerum itaque, facilis, mollitia ducimus eaque repellendus!

[READMORE](#)

---

[Live Demo](#)

## СТИЛИЗУЕМ БЛОК

Для расположения блоков даты и кнопки `Readmore` по краям родителя мы использовали свойство `float`:

```
1  .article::after {
2      content: "";
3      display: block;
4      clear: both;
5  }
6
7  .article__date,
8  .article__content {
9      float: left;
10 }
11
12 .article__showmore {
13     margin-top: 110px;
14     float: right;
15 }
```

## ПЕРЕПИШЕМ НА FLEXBOX

Давайте перепишем стили и используем уже знакомые flexbox:

```
1  .article {  
2    display: flex;  
3  }  
4  
5  .article__showmore {  
6    margin-top: 110px;  
7  }
```



---

# ИЗБАВИЛИСЬ ОТ CLEARFIX

Код стал намного проще и мы избавились от хака clearfix. Однако что-то пошло не так:

20  
JUN

## The Basics of Binary and Hex, Explained in Video Form

Lorem ipsum dolor sit amet, consectetur adipisicing elit. Deleniti cum error deserunt! Nostrum, architecto! Temporibus ullam inventore totam ea necessitatibus repellat placeat ad saepe laboriosam natus error vitae numquam unde eligendi cum maiores rerum itaque, facilis, mollitia ducimus eaque repellendus!

READMORE

---

[Live Demo](#)

# МАКСИМАЛЬНО ВПРАВО

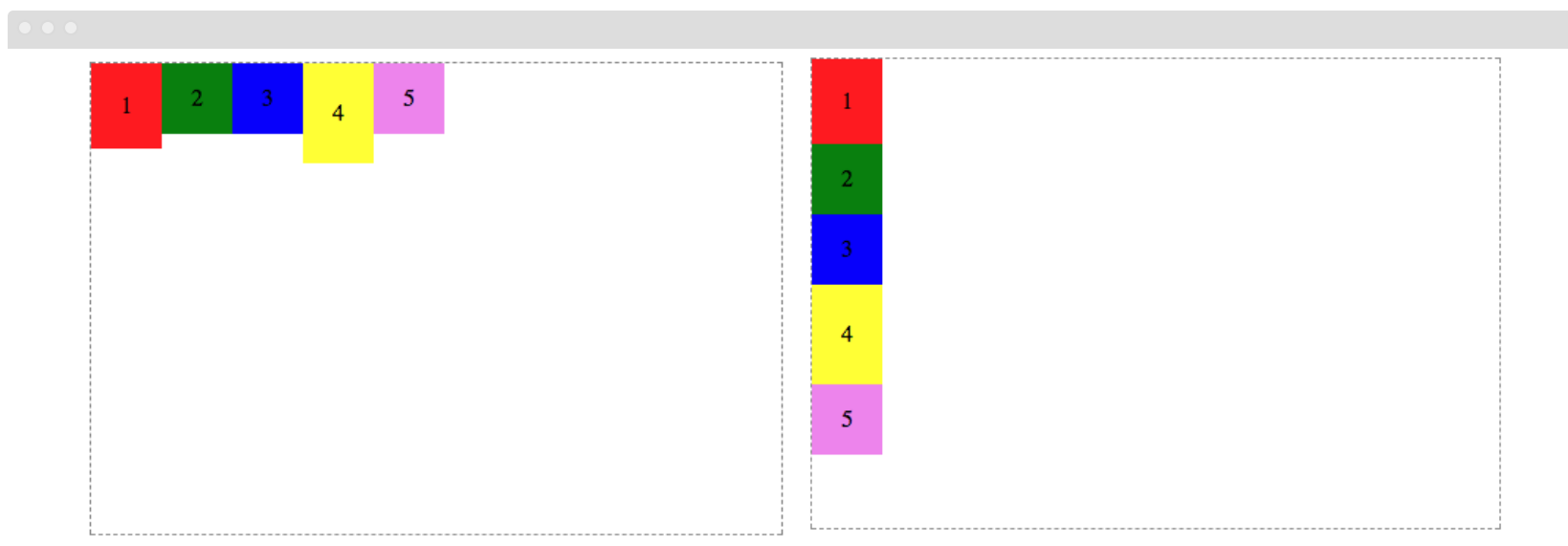
Кнопка вытянулась, потому что не указано, как будут выравниваться flex-элементы по вертикали. За это отвечает свойство `align-items`.

Возможные значения:

- `flex-start`
- `flex-end`
- `center`
- `baseline`
- `stretch`

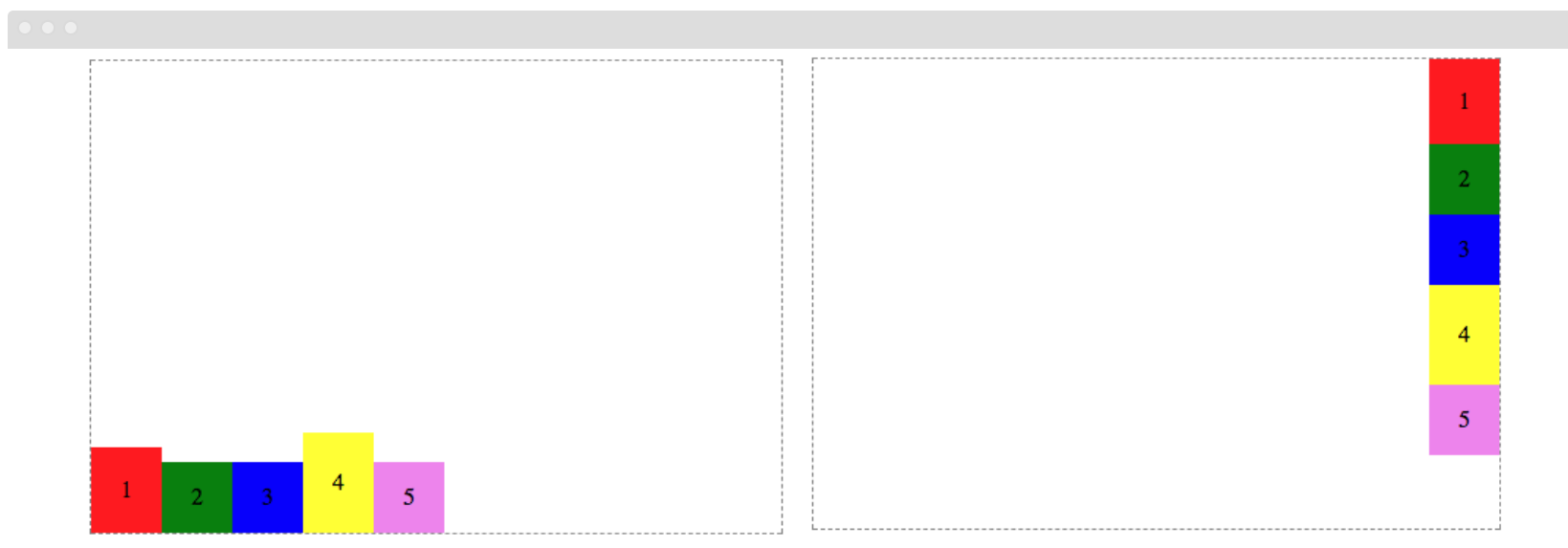
# flex-start

При `align-items: flex-start` браузер переместит элементы в начало flex-контейнера:



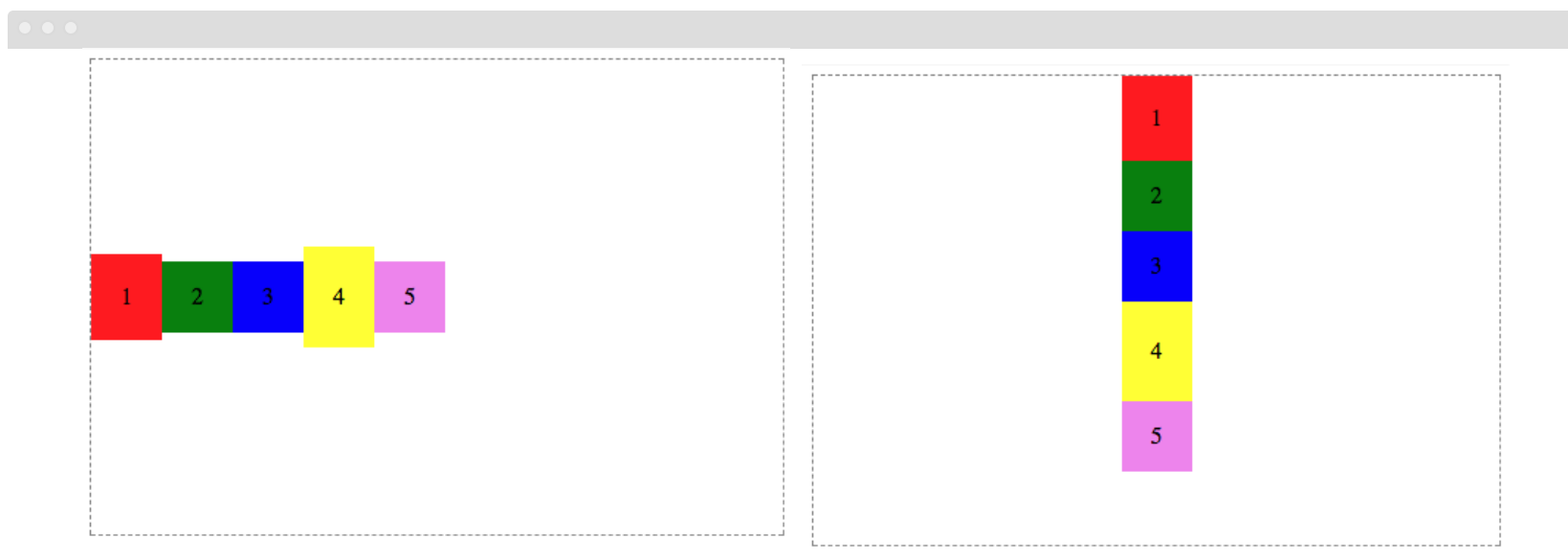
# flex-end

Значение `flex-end` располагает flex-элементы в конце flex-контейнера:



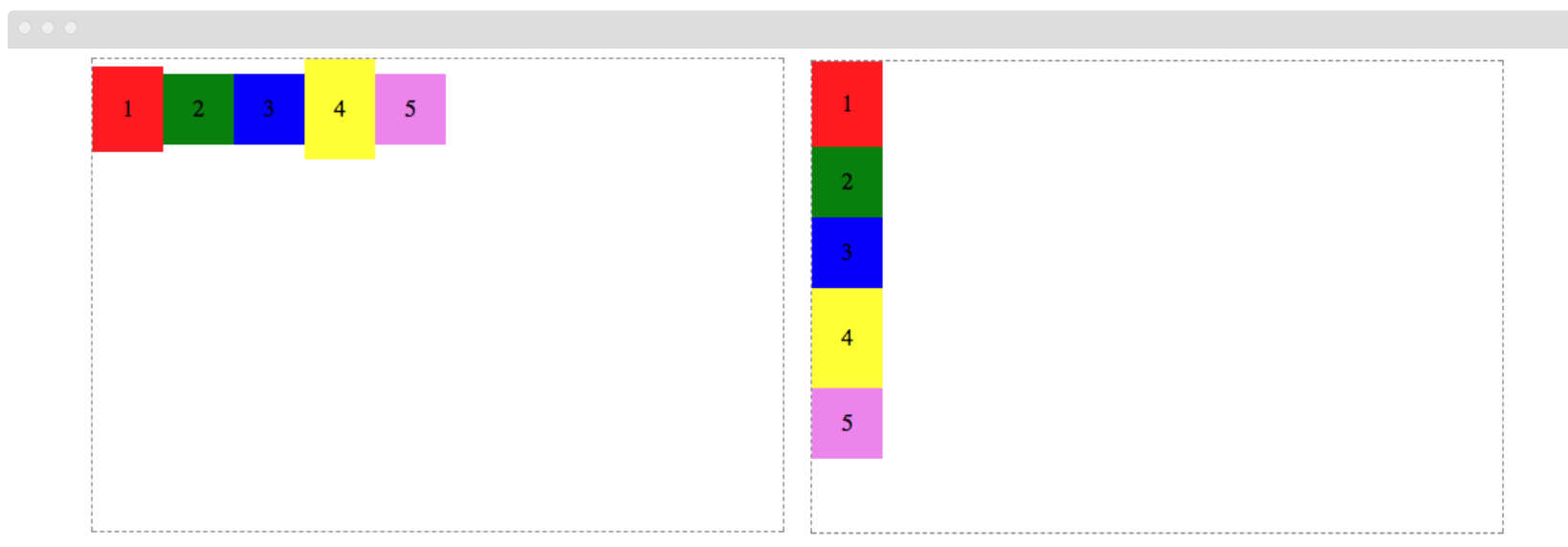
# center

Значение `center` отцентрирует элементы по дополнительной оси:



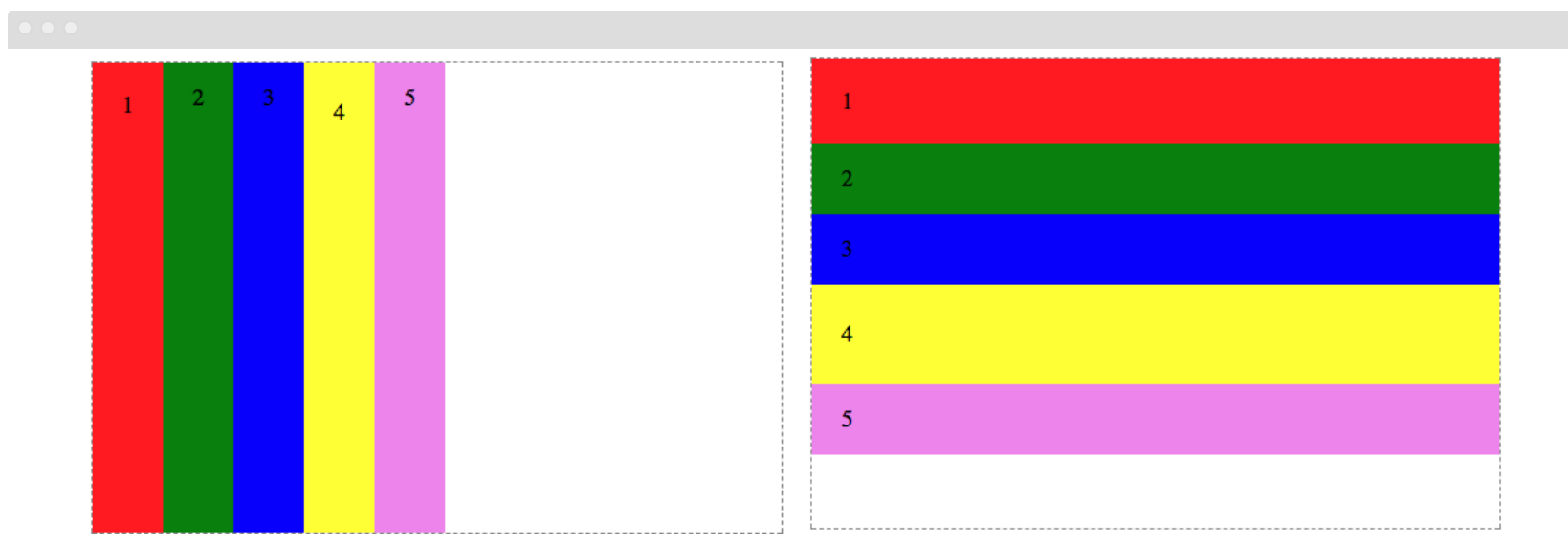
# baseline

Значение `baseline` выравнивает элементы по базовой линии:



# stretch

При этом значении flex-элементы растягиваются по дополнительной оси:



`stretch` является значением по умолчанию и его можно не указывать. Именно поэтому наша кнопка и оказалась растянутой.

## ПОПРАВИМ КНОПКУ

Чтобы привести верстку к нужному виду, допишем в код нашего примера:

```
1  .article {  
2    display: flex;  
3    align-items: flex-start;  
4  }
```



---

# КНОПКА В НОРМЕ



## 20

JUN

### **The Basics of Binary and Hex, Explained in Video Form**

Lorem ipsum dolor sit amet, consectetur adipisicing elit. Deleniti cum error deserunt! Nostrum, architecto! Temporibus ullam inventore totam ea necessitatibus repellat placeat ad saepe laboriosam natus error vitae numquam unde eligendi cum maiores rerum itaque, facilis, mollitia ducimus eaque repellendus!

READMORE

---

[Live Demo](#)

# justify-content

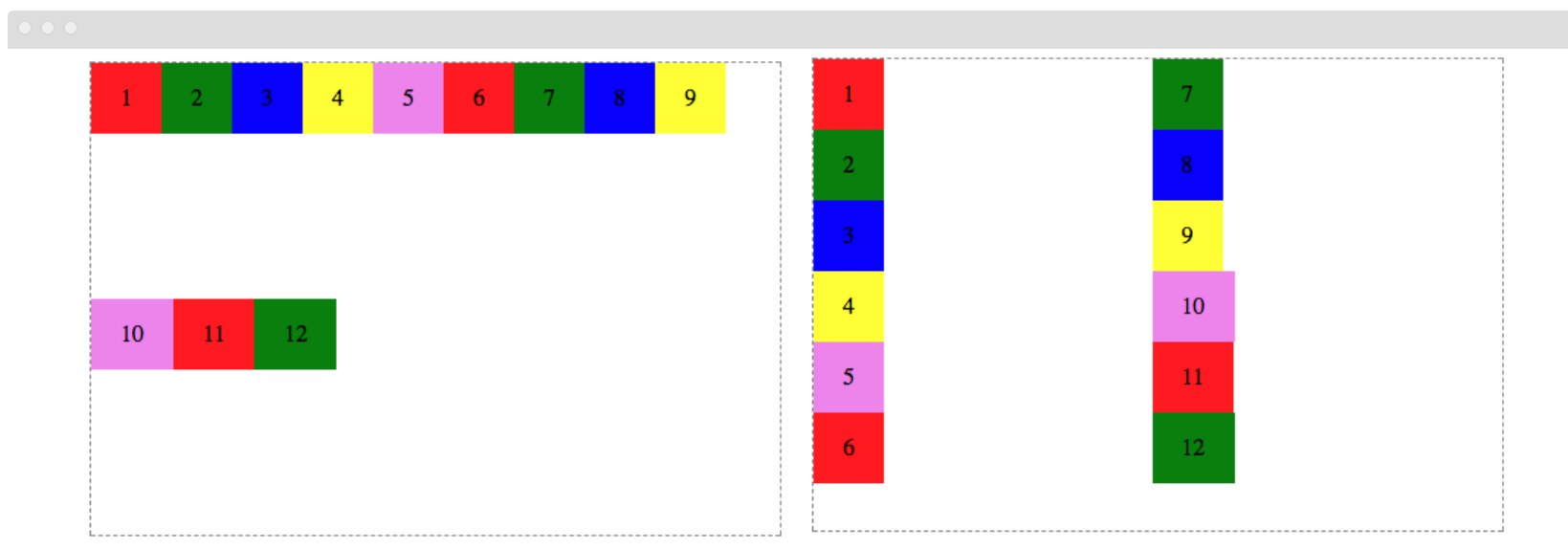
Помимо выравнивания по вертикали, мы можем указать, как браузер будет распределять свободное место между элементами и вокруг элементов в контейнере. За это отвечает свойство `justify-content`.

Возможные значения:

- `flex-start`
- `flex-end`
- `center`
- `space-between`
- `space-around`
- `space-evenly`

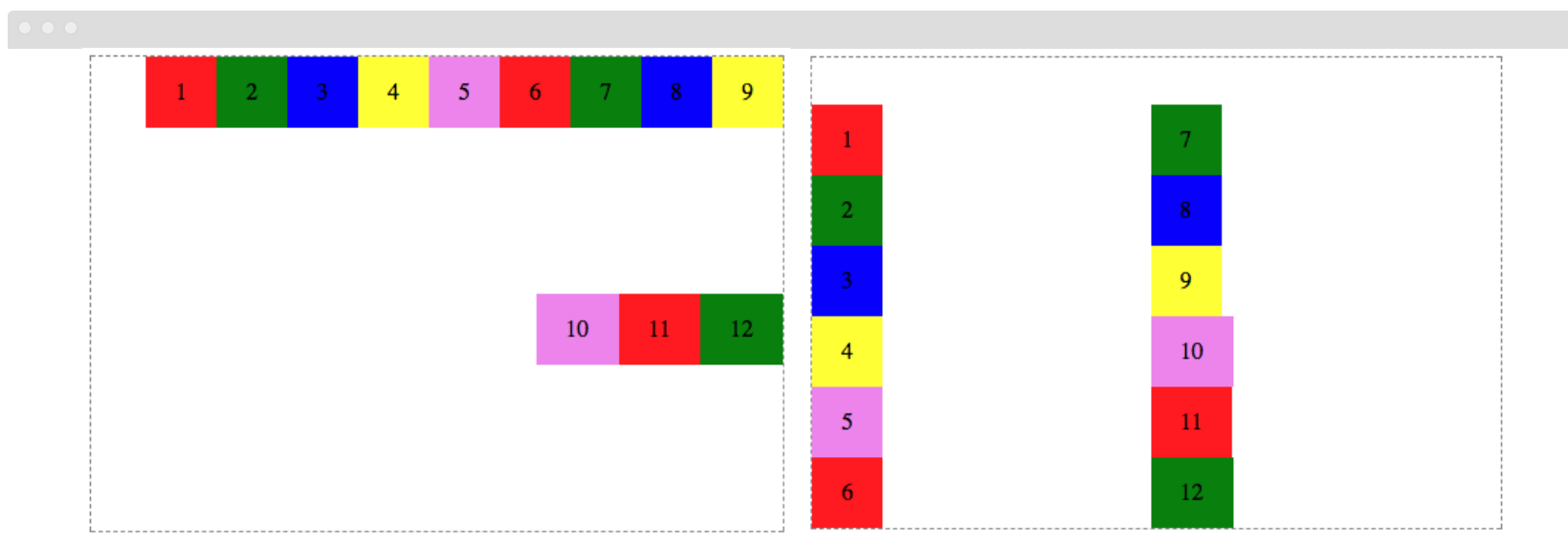
# flex-start

Значение по умолчанию. При этом flex-элементы выстраиваются в начале flex-контейнера:



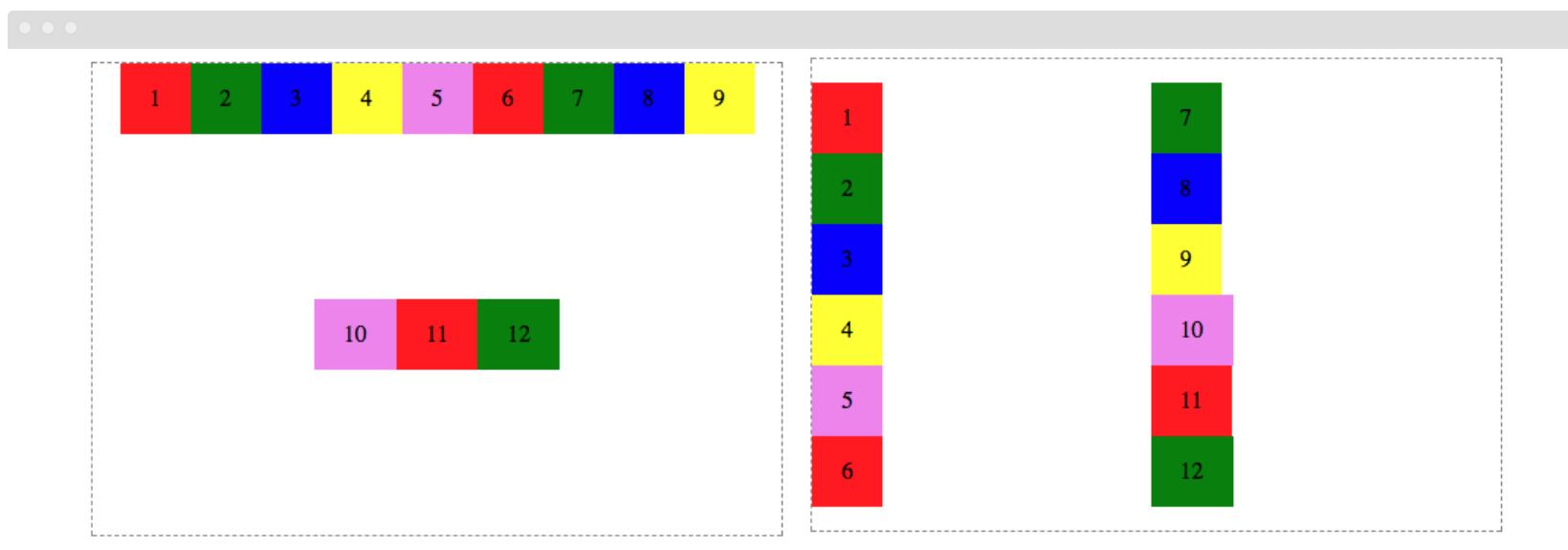
# flex-end

При этом значении flex-элементы выстраиваются в конце flex-контейнера:



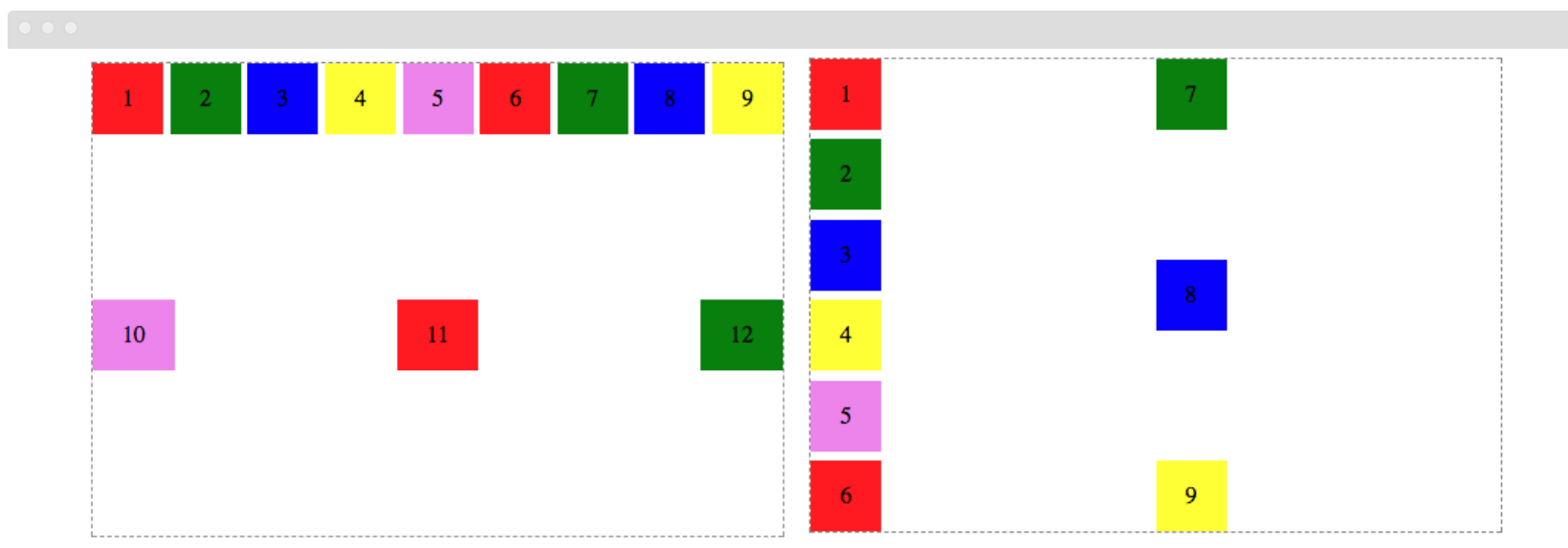
# center

Значение выравнивает элементы по центру контейнера:



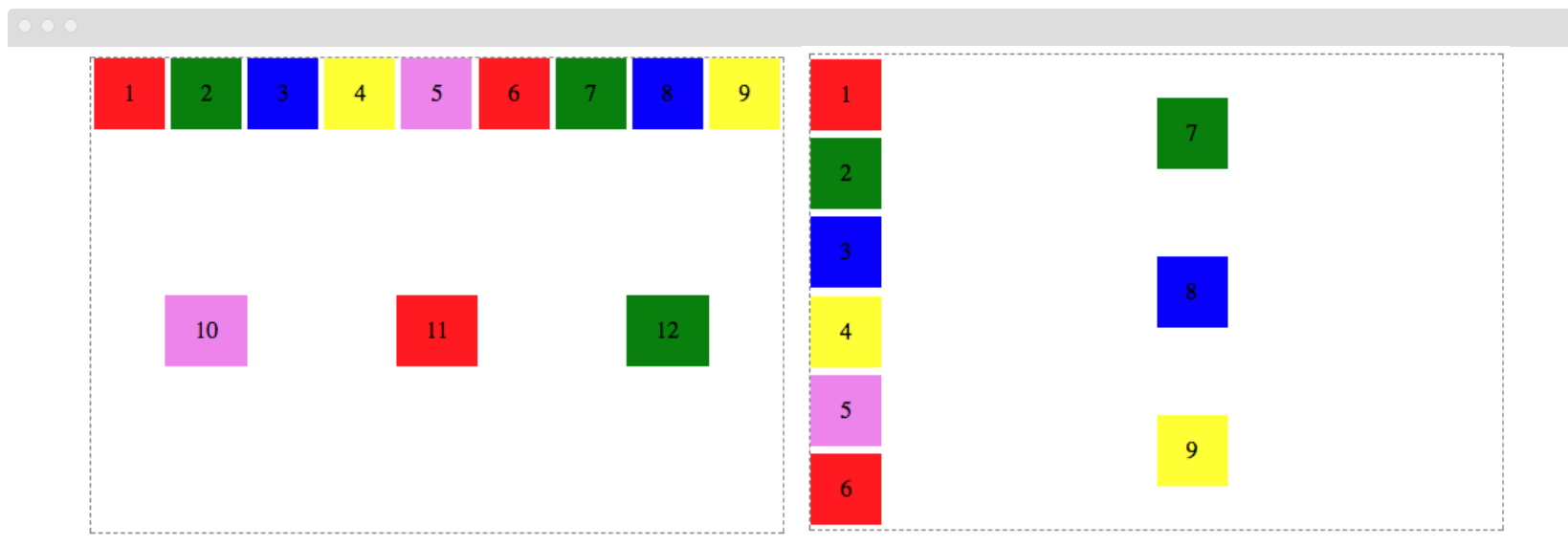
# space-between

располагает элементы равномерно так, что первый и последний находятся вплотную к краям.



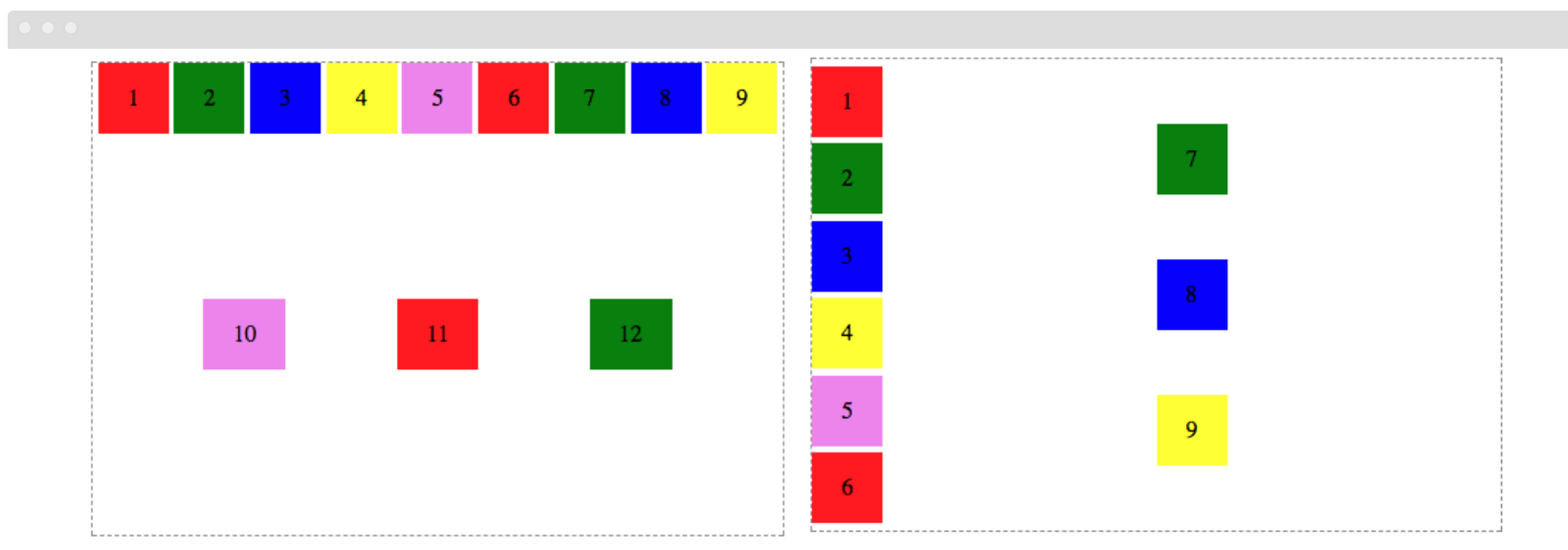
# space-around

Располагает элементы равномерно, при этом расстояние между ними одинаково, а расстояние от первого/последнего элемента до краев равняется половине интервала между элементами.



# space-evenly

Располагает элементы равномерно, при этом расстояния между ними, а также от первого/последнего элемента до краев одинаковы.



[Интерактивная песочница](#)



## ЭЛЕМЕНТЫ ПО КРАЯМ РОДИТЕЛЯ

До того момента, когда мы убрали `float`, дата и кнопка располагались по краям родительского контейнера. Но как только мы добавили `display: flex`, элементы перестали прижиматься к границам. Нужно это исправить. Зададим свойство `justify-content`:

```
1  .article {  
2    display: flex;  
3    align-items: flex-start;  
4    justify-content: space-between;  
5  }  
6  
7  .article__showmore {  
8    margin-top: 110px;  
9  }
```

---

# ВЗГЛЯНЕМ НА МАКЕТ

20  
JUN

## The Basics of Binary and Hex, Explained in Video Form

Lorem ipsum dolor sit amet, consectetur adipisicing elit. Deleniti cum error deserunt! Nostrum, architecto! Temporibus ullam inventore totam ea necessitatibus repellat placeat ad saepe laboriosam natus error vitae numquam unde eligendi cum maiores rerum itaque, facilis, mollitia ducimus eaque repellendus!

[READMORE](#)

---

Верстка макету соответствует.

# ИЗМЕНЯЕМ КОЛИЧЕСТВО КОНТЕНТА

**18**  
**JUN**

## A Simple Software Update

Lorem ipsum dolor sit amet, consectetur adipisicing elit. Deleniti cum error deserunt! Nostrum, architecto!

[READMORE](#)

---

[Live Demo](#)

## РАСХОЖДЕНИЯ С МАКЕТОМ

Значение `110px` для `margin-top` рассчитано с учетом того, что блок имел высоту `260px`. Но теперь кнопка находится заметно ниже блока с контентом. Для решения этой задач и нам поможет свойство `align-self`.

## align-self

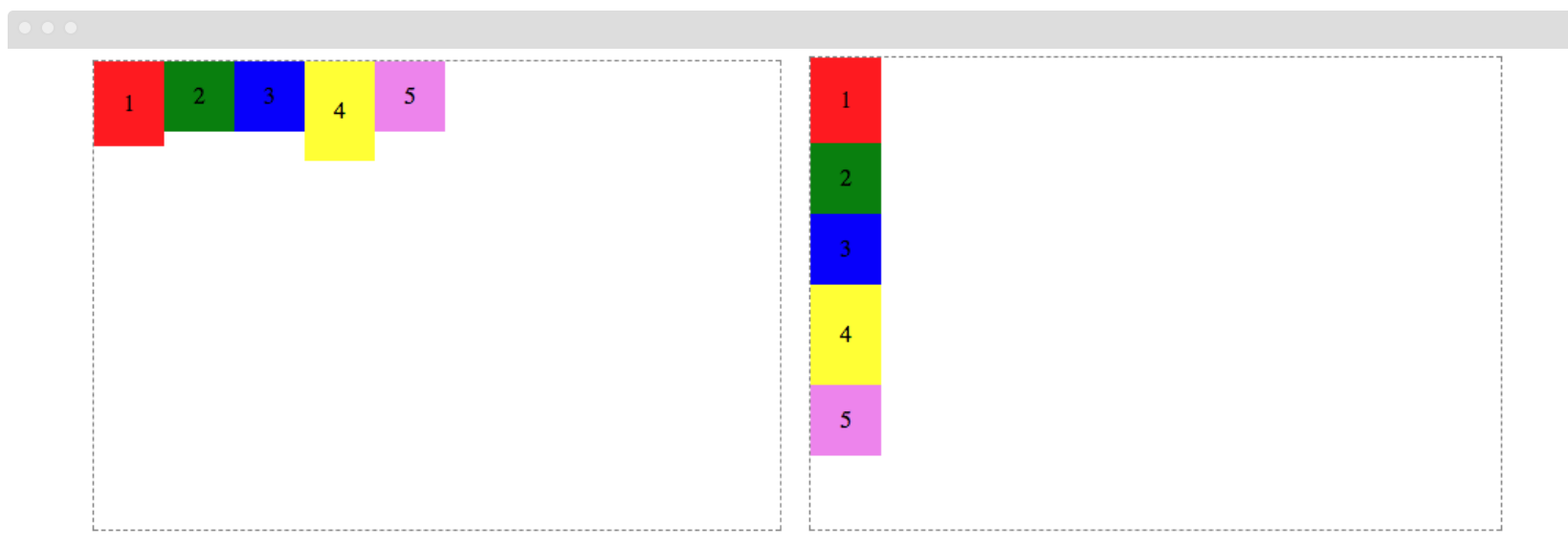
Свойство `align-self` очень похоже на `align-items`. Оно также выравнивает элемент вдоль второстепенной оси. Но есть отличие. С помощью свойства `align-self` можно переопределить `align-item` для отдельного элемента.

Доступные значения:

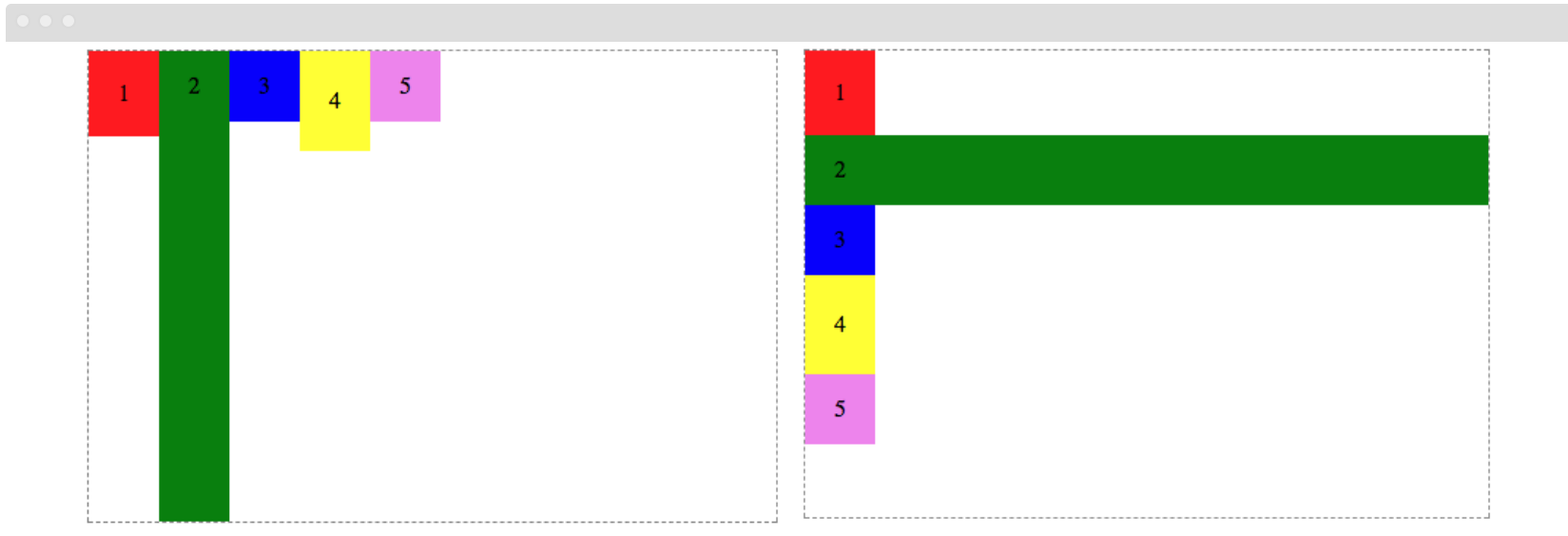
- `auto`
- `stretch`
- `flex-start`
- `flex-end`
- `center`
- `baseline`

# auto

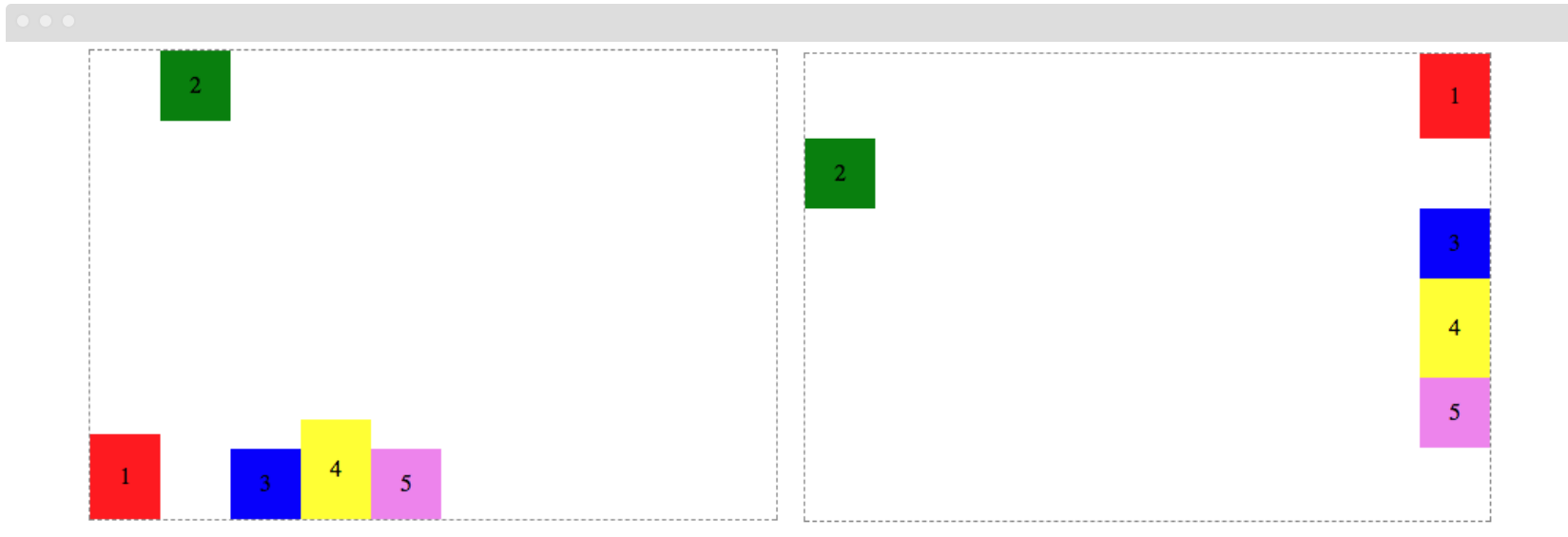
Значение по умолчанию.



# stretch

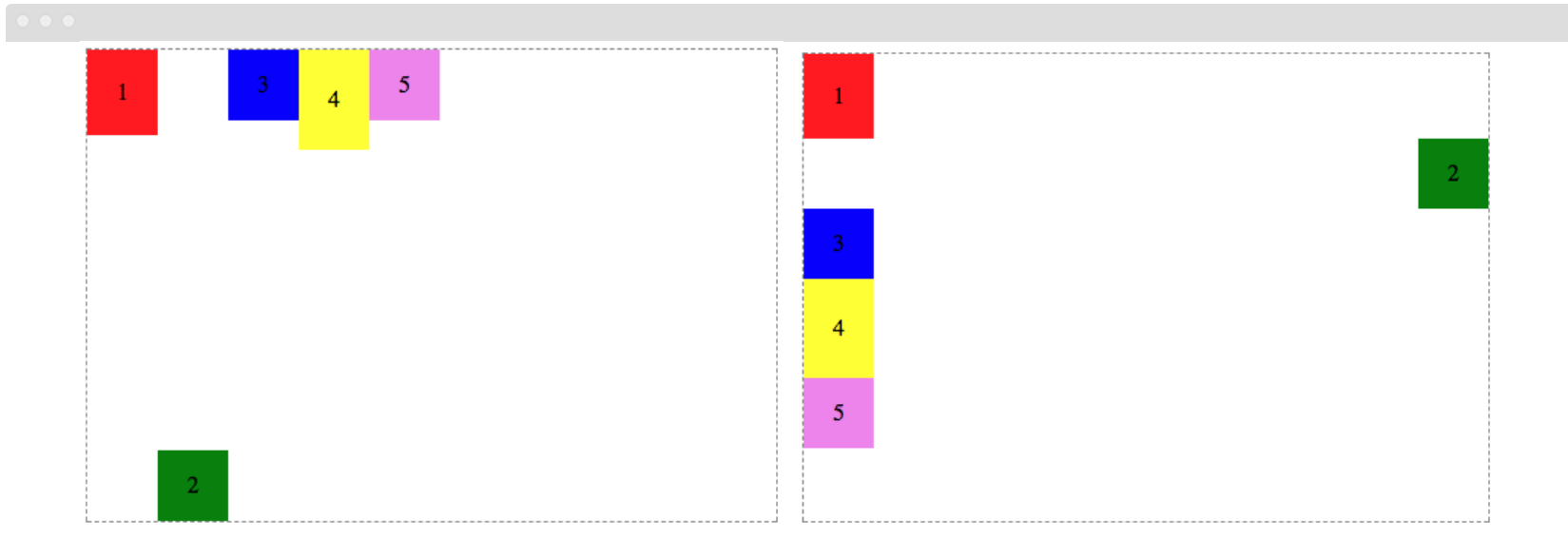


# flex-start



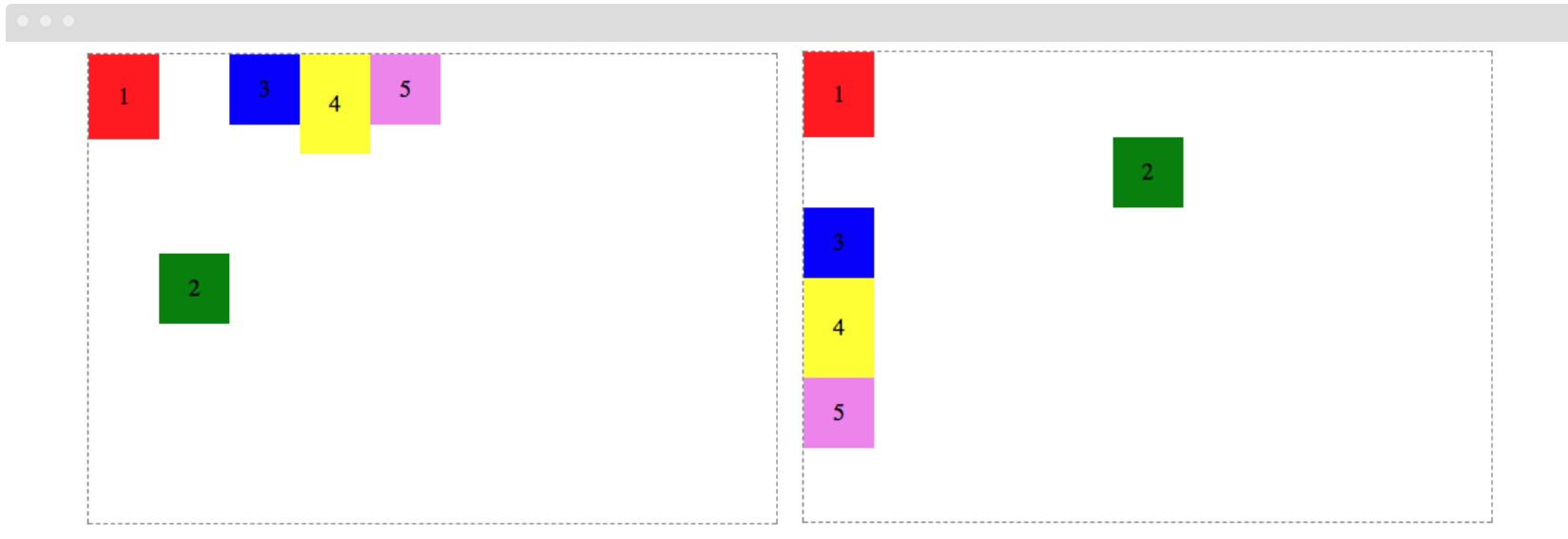


# flex-end

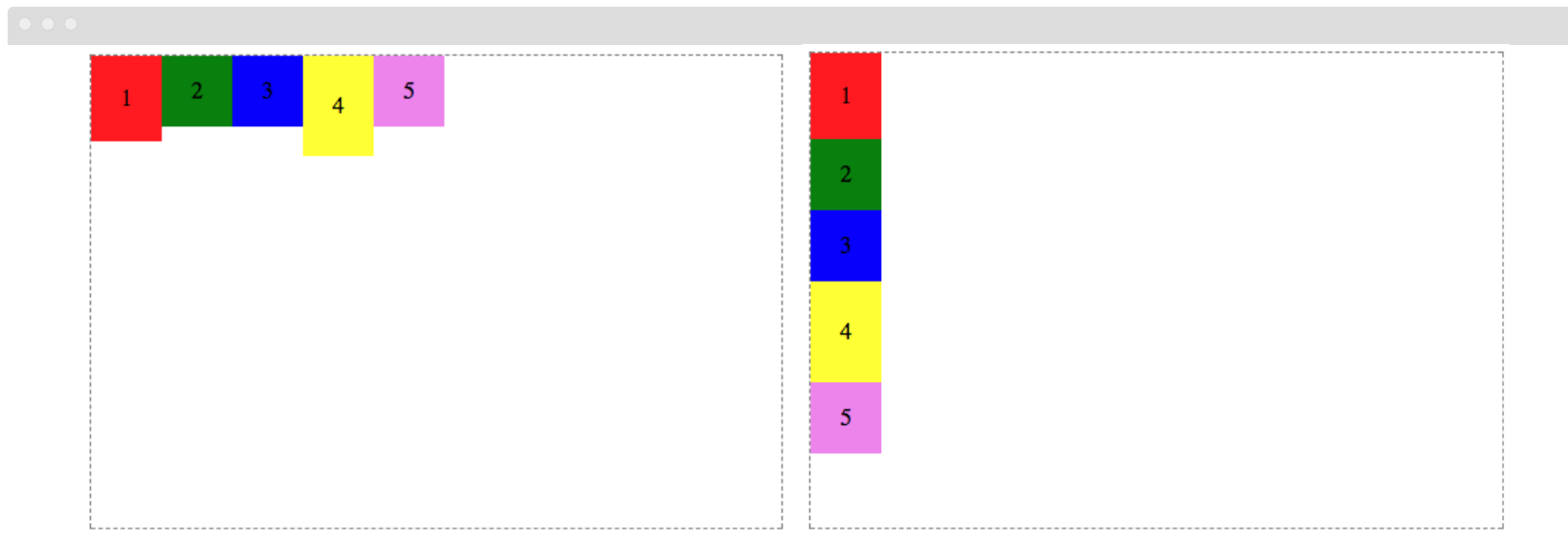


---

# center



# baseline



[Интерактивная песочница](#)

# КНОПКА ВСЕГДА ПО ЦЕНТРУ

Применим к кнопке свойство `align-self` со значением `center`.  
Удалим из класса `.article__showmore` свойство `margin-top` и добавим свойство `align-self`:

```
1 | .article__showmore {  
2 |     align-self: center;  
3 | }
```

[Live Demo](#)

---

# ВСЕ РАБОТАЕТ

**18**  
JUN

## A Simple Software Update

Lorem ipsum dolor sit amet, consectetur adipisicing elit. Deleniti cum error deserunt! Nostrum, architecto!

[READMORE](#)

---

# ДОБАВИМ БОЛЬШЕ КОНТЕНТА

20  
JUN

## The Basics of Binary and Hex, Explained in Video Form

Lorem ipsum dolor sit amet, consectetur adipisicing elit. Deleniti cum error deserunt! Nostrum, architecto! Temporibus ullam inventore totam ea necessitatibus repellat placeat ad saepe laboriosam natus error vitae numquam unde eligendi cum maiores rerum itaque, facilis, mollitia ducimus eaque repellendus!

[READMORE](#)

---

Теперь кнопка всегда будет располагаться по центру относительно блока с КОНТЕНТОМ.



# ИТОГИ

# ГРАФИКА

- Перед загрузкой изображения, браузер анализирует его размеры (3 случая: оставляет как есть, уменьшает, растягивает).
- Формула расчета пикселей:  $P_{px} = CSS_{px} * DPR$ , где DPR (Device Pixel Ratio) — коэффициент соотношения физического пикселя к виртуальному.
- Вариант решения: для всех экранов загружать иконки большего размера, а в CSS указать нужный размер.



# ИКОНОЧНЫЙ ШРИФТ

- Это шрифт, в котором символами являются иконки. Можно работать как с текстом, применять свойства `font-size`, `color` и прочие.
- Есть сайты, где можно собрать свой шрифт (например, <http://fontello.com/>).
- Недостаток: иконки являются текстом, несемантично.

---

# SVG

- Масштабируемая векторная графика, имеет все преимущества иконочных шрифтов, но при этом является графикой.
- Недостаток внешнего файла: не можем изменять цвет иконки, она становится как будто за стеклом и до нее не достучаться.
- Как встроить SVG в HTML? Открываем файл \*.svg в любом текстовом редакторе, копируем код и вставляем в HTML.
- Атрибут `fill` используем для заливки иконки. Удаляем его, если он задан инлайн, и вставляем его в CSS.

## ОСОБЕННОСТИ ВЕРСТКИ ФОРМ НА МОБИЛЬНЫХ УСТРОЙСТВАХ

- В браузере есть CSS-правила, которые определены в операционной системе.
- Свойство `appearance` делает любой элемент выглядящим как элемент формы со стандартными стилями операционной системы. Используем значение `none`, если это нужно убрать.
- Свойство `appearance` не поддерживается браузерами. Используем вендорные префиксы `-webkit` и `-moz`.

# ОТОБРАЖЕНИЕ ТЕЛЕФОНА СИНИМ ЦВЕТОМ НА IOS

- Разработчики iOS позаботились о пользователях: номер телефона становится ссылкой с протоколом `tel:`.
- Можно запретить мобильному браузеру распознавать блоки с номерами телефонов. Есть специальный метатег для секции `head`: `<meta name="format-detection" content="telephone=no">`
- Но лучше добавить в CSS правило для селектора .

# ВЫРАВНИВАНИЕ FLEXBOX-ЭЛЕМЕНТОВ

- Свойство `align-items` задаёт как будут выравниваться flex-элементы по дополнительной оси.
- Свойство `justify-content` указывает, как браузер будет распределять свободное место между элементами и вокруг элементов в контейнере.
- С помощью свойства `align-self` можно переопределить `align-item` для отдельного элемента.



**НЕТОЛОГИЯ**  
университет интернет-профессий

**Задавайте вопросы и напишите отзыв о лекции!**

**СЕМЕН БОЙКО**



[simonderus@gmail.com](mailto:simonderus@gmail.com)



[fb.me/simonboycko](https://fb.me/simonboycko)