САНКТ-ПЕТЕРБУРГСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО

Дисциплина: Фронд-энд разработка

Отчет

Практическая работа 1

Выполнил:
Прохоров Николай

Группа: К33402

Проверил:
Добряков Д. И.

Санкт-Петербург

2021 г.

Задача

Освоить практические навыки работы с CSS Flexbox, CSS Grid, выполняя 2 игры:

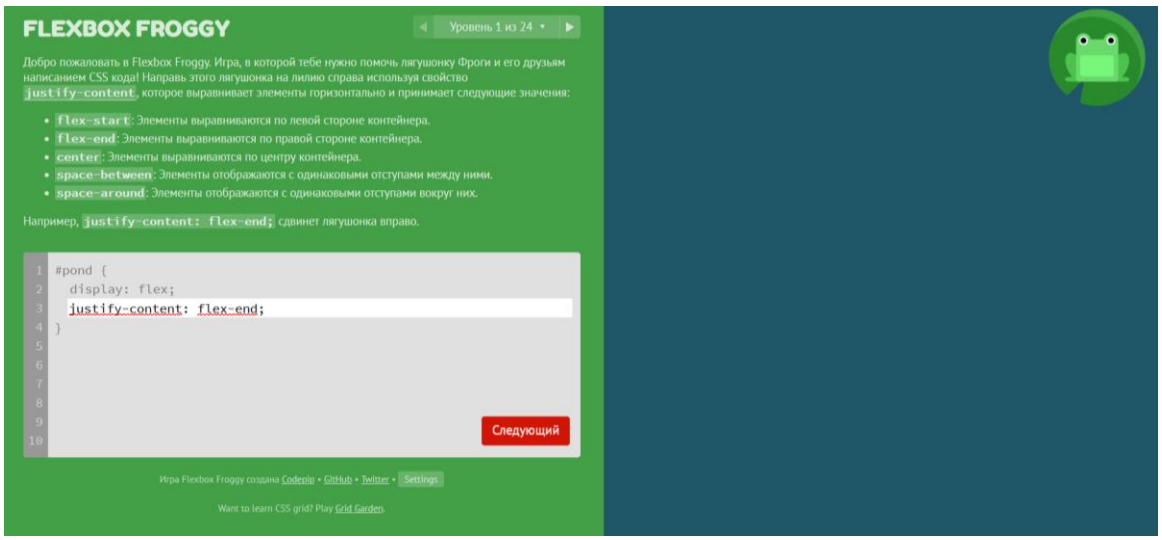
https://flexboxfroggy.com/#ru
https://cssgridgarden.com/#ru

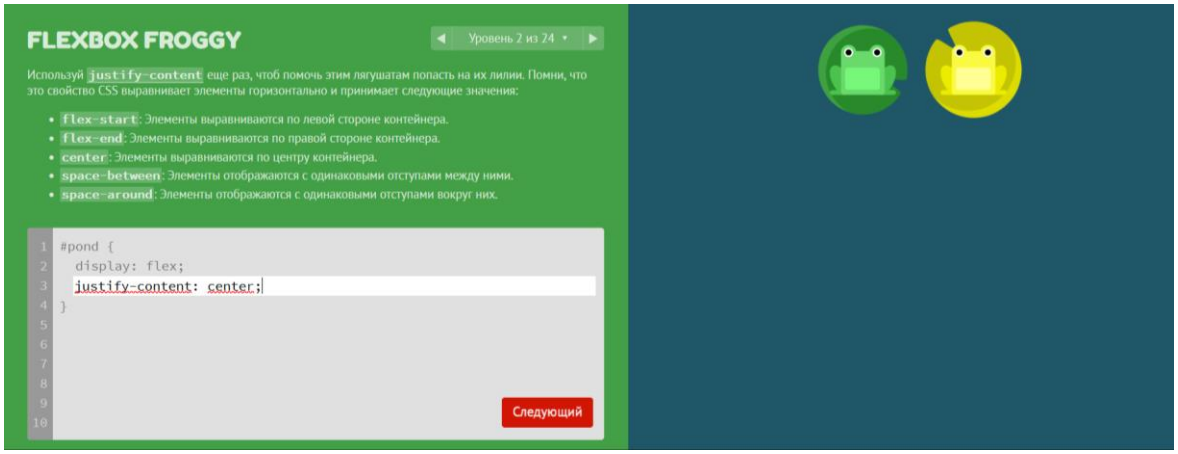
Ход работы

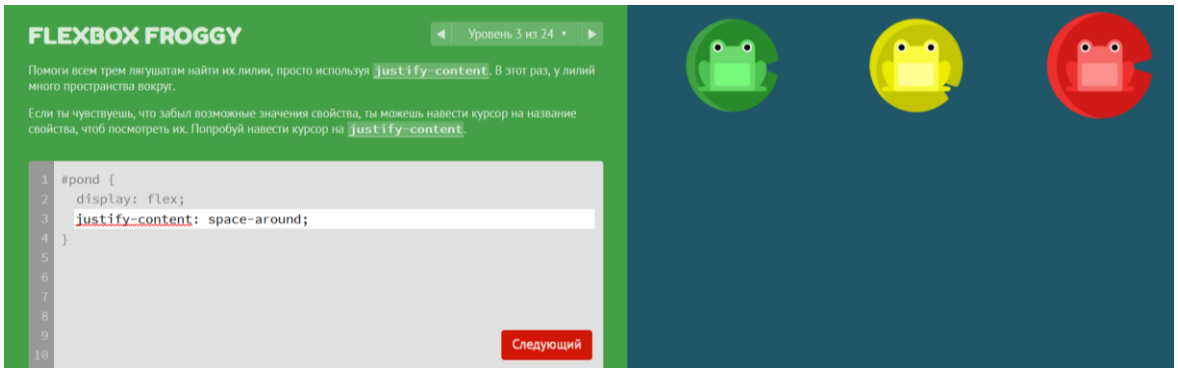
Проходим постепенно все 24 уровня Flexbox Froggy, познаем новые свойства.

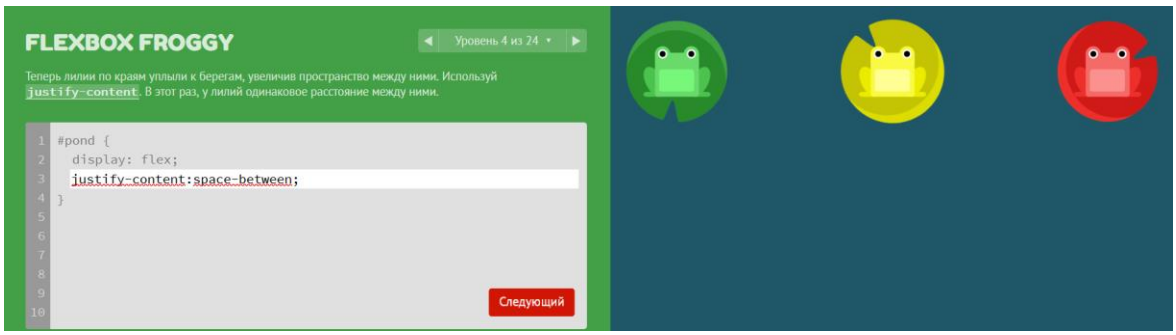
За эти 24 уровня мы изучили следующие свойства:

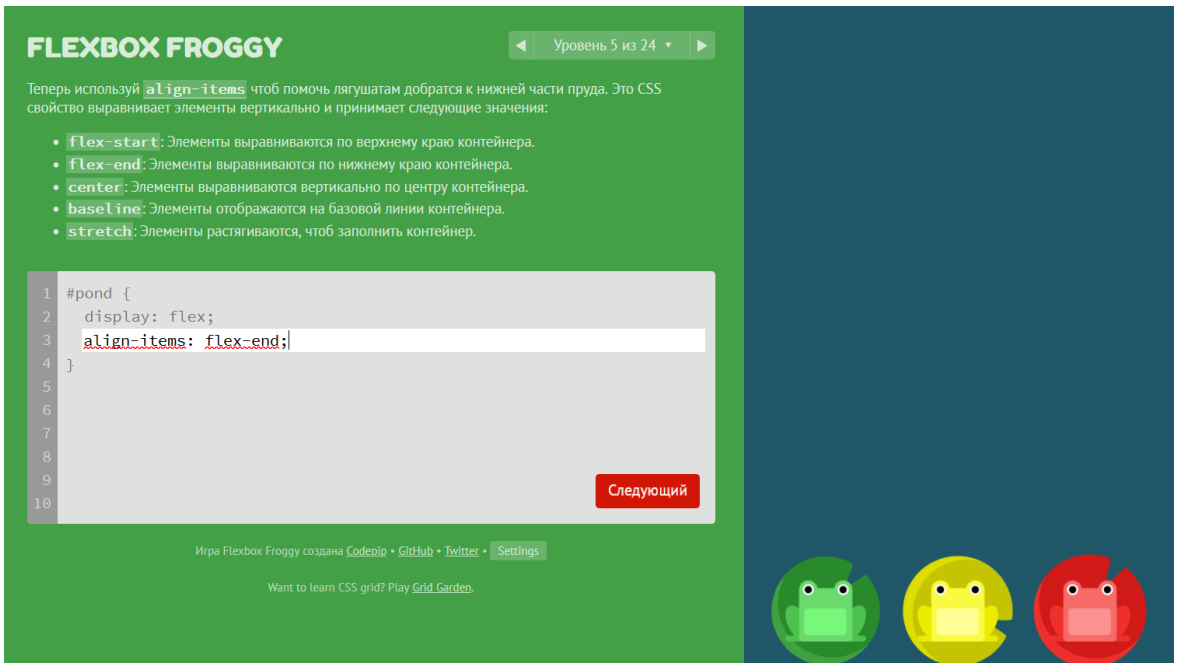
- justify-content
- align-items
- flex-direction
- order
- align-self
- flex-wrap
- flex-flow
- align-content

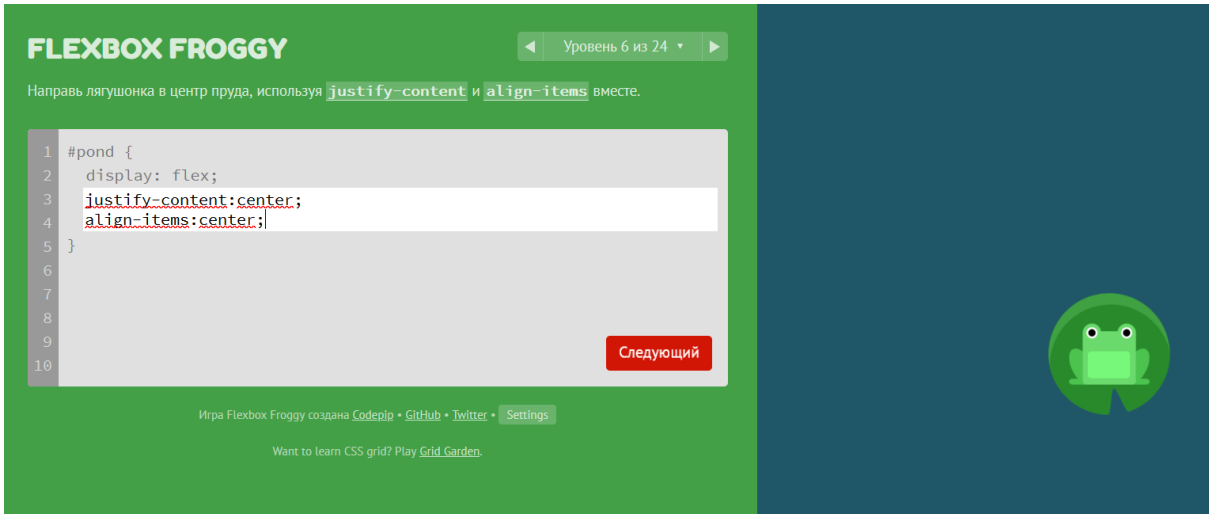





FLEXBOX FROGGY
Уровень 3 из 24
Помоги всем трем лягушатам найти их лилии, просто используя justify-content. В этот раз, у лилий много пространства вокруг.
Если ты чувствуешь, что забыл возможные значения свойства, ты можешь навести курсор на название свойства, чтоб посмотреть их. Попробуй навести курсор на justify-content.
#pond {
display: flex;
justify-content: space-around;
}
Следующий


FLEXBOX FROGGY
Уровень 4 из 24
Теперь лилии по краям уплыли к берегам, увеличив пространство между ними. Используй justify-content. В этот раз, у лилий одинаковое расстояние между ними.
#pond {
display: flex;
justify-content:space-between;
}
Следующий


FLEXBOX FROGGY
Уровень 5 из 24
Теперь используй align-items чтоб помочь лягушатам добрался к нижней части пруда. Это CSS свойство выравнивает элементы вертикально и принимает следующие значения:
flex-start: Элементы выравниваются по верхнему краю контейнера.
flex-end: Элементы выравниваются по нижнему краю контейнера.
center: Элементы выравниваются вертикально по центру контейнера.
baseline: Элементы отображаются на базовой линии контейнера.
stretch: Элементы растягиваются, чтоб заполнить контейнер.
#pond {
display: flex;
align-items: flex-end;
}
Следующий
Игра Flexbox Froggy создана Codepip • GitHub • Twitter • Settings
Want to learn CSS grid? Play Grid Garden.


FLEXBOX FROGGY
Уровень 6 из 24
Направь лягушонка в центр пруда, используя justify-content и align-items вместе.
#pond {
display: flex;
justify-content:center;
align-items:center;
}
Следующий
Игра Flexbox Froggy создана Codepip • GitHub • Twitter • Settings
Want to learn CSS grid? Play Grid Garden.

FLEXBOX FROGGY

Уровень 7 из 24

Лягушатам снова нужно пересечь пруд. В этот раз к лилиям, с достаточно большим пространством вокруг них. Используй комбинацию `justify-content` и `align-items`.

```
#pond {
  display: flex;
  justify-content: space-around;
  align-items: flex-end;
}
```

Следующий

Игра Flexbox Froggy создана Codepip • GitHub • Twitter • Settings

Want to learn CSS grid? Play Grid Garden.

FLEXBOX FROGGY

Уровень 8 из 24

Лягушатам нужно выстроиться в порядке их лилий, используя `flex-direction`. Это CSS свойство задает направление, в котором будут расположены элементы в контейнере, и принимает следующие значения:

- `row`: элементы размещаются по направлению текста.
- `row-reverse`: элементы отображаются в обратном порядке к направлению текста.
- `column`: элементы располагаются сверху вниз.
- `column-reverse`: элементы располагаются снизу вверх.

```
#pond {
  display: flex;
  flex-direction: row-reverse;
}
```

Следующий

FLEXBOX FROGGY

Уровень 9 из 24

Помоги лягушатам расположиться на своих лилиях используя `flex-direction`. Это CSS свойство задает направление, в котором будут расположены элементы в контейнере и принимает следующие значения:

- `row`: Элементы размещаются по направлению текста.
- `row-reverse`: Элементы отображаются в обратном порядке к направлению текста.
- `column`: Элементы располагаются сверху вниз.
- `column-reverse`: Элементы располагаются снизу вверх.

```
#pond {
  display: flex;
  flex-direction:column;
}
```

Следующий

FLEXBOX FROGGY

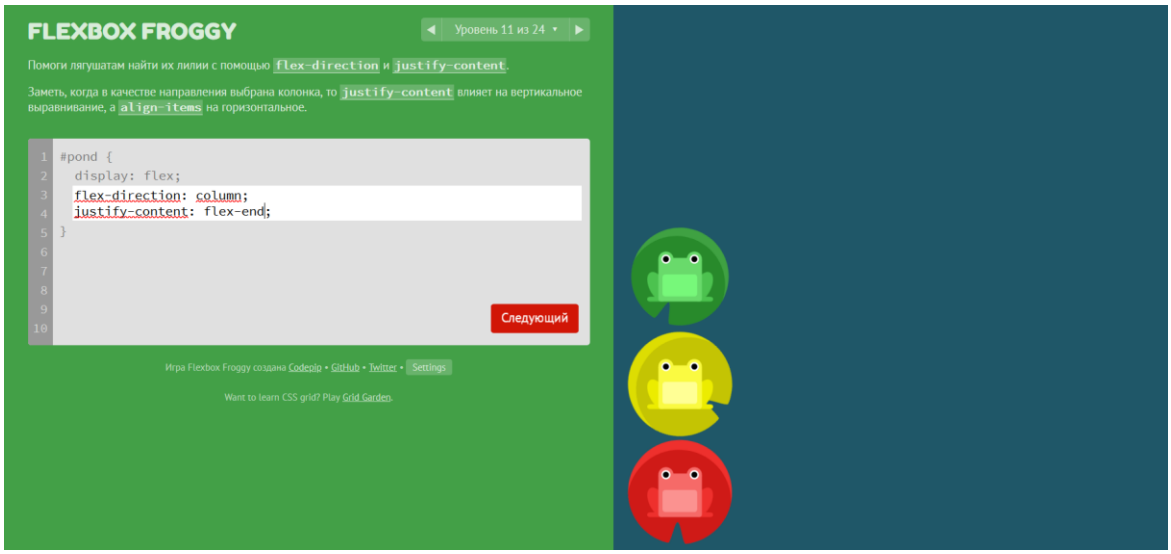
Уровень 10 из 24

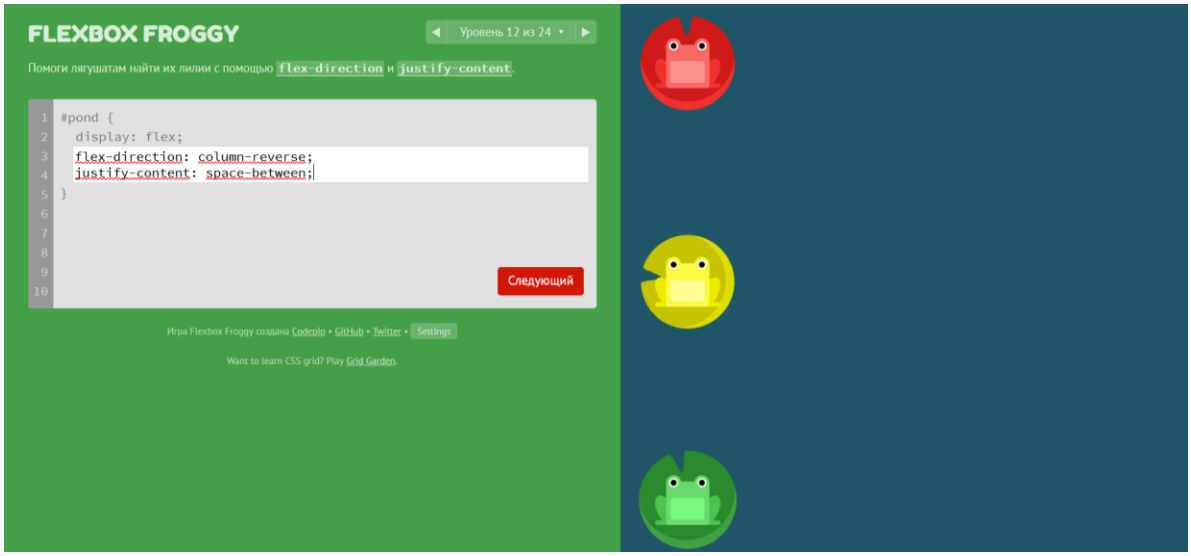
Помоги лягушатам попасть на свои лилии. Хоть и кажется, что они почти на своих местах, все же придется применить и `flex-direction` и `justify-content`, чтоб поместить их на свои лилии.

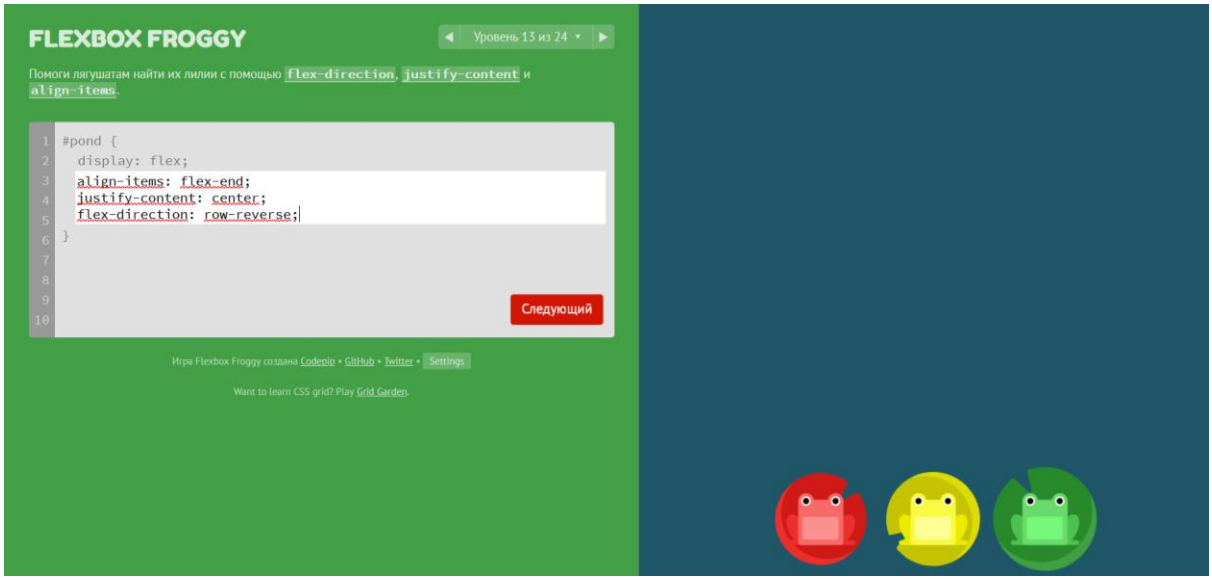
Заметь, что когда ты устанавливаешь направление в обратном порядке для ряда или колонки, начало (start) и конец (end) тоже меняются местами.

```
#pond {
  display: flex;
  flex-direction: row-reverse;
  justify-content: flex-end;
}
```

Следующий


FLEXBOX FROGGY
Уровень 11 из 24
Помоги лягушатам найти их лилии с помощью flex-direction и justify-content.
Заметь, когда в качестве направления выбрана колонка, то justify-content влияет на вертикальное выравнивание, а align-items на горизонтальное.
#pond {
display: flex;
flex-direction: column;
justify-content: flex-end;
}
Следующий
Игра Flexbox Froggy создана Codepip • GitHub • Twitter • Settings
Want to learn CSS grid? Play Grid Garden.


FLEXBOX FROGGY
Уровень 12 из 24
Помоги лягушатам найти их лилии с помощью flex-direction и justify-content.
#pond {
display: flex;
flex-direction: column-reverse;
justify-content: space-between;
}
Следующий
Игра Flexbox Froggy создана Codepip • GitHub • Twitter • Settings
Want to learn CSS grid? Play Grid Garden.


FLEXBOX FROGGY
Уровень 13 из 24
Помоги лягушатам найти их лилии с помощью flex-direction, justify-content и align-items.
#pond {
display: flex;
align-items: flex-end;
justify-content: center;
flex-direction: row-reverse;
}
Следующий
Игра Flexbox Froggy создана Codepip • GitHub • Twitter • Settings
Want to learn CSS grid? Play Grid Garden.

FLEXBOX FROGGY

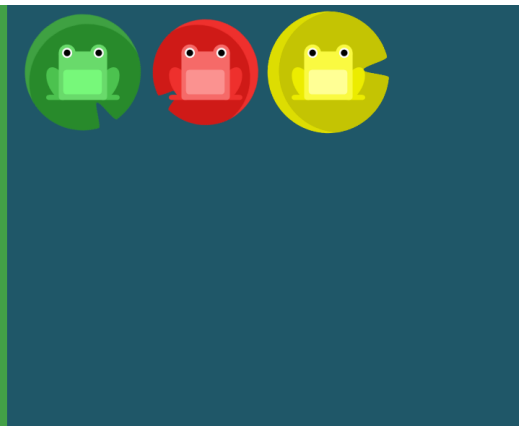
Уровень 14 из 24

Иногда изменения порядка отображения элементов в контейнере недостаточно. В таких случаях мы можем применить свойство `order` для конкретных элементов. По умолчанию, значение этого свойства у элементов равно 0, но мы можем задать положительное или отрицательное целое число этому свойству.

Используй свойство `order`, чтоб разместить лягушат на своих лилиях.

```
#pond {
  display: flex;
}

.yellow {
  order: 1;
}
```

Следующий



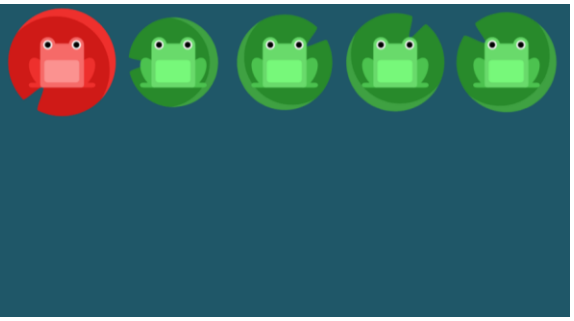
FLEXBOX FROGGY

Уровень 15 из 24

Используй свойство `order`, чтоб отправить красного лягушонка на его лилию.

```
#pond {
  display: flex;
}

.red {
  order:-1;
}
```

Следующий



FLEXBOX FROGGY

Уровень 16 из 24

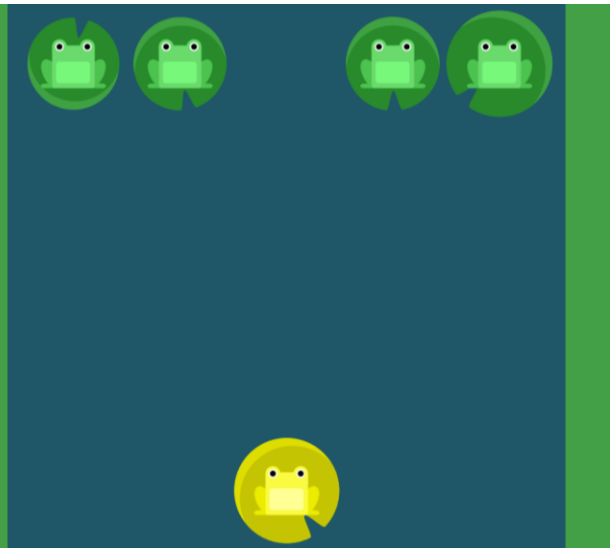
Еще одно свойство, которое ты можешь применить к определенному элементу это `align-self`. Это свойство принимает те же значения, что и `align-items`.

```
#pond {
  display: flex;
  align-items: flex-start;
}

.yellow {
  align-self: flex-end;
}
```

Следующий

Игра Flexbox Froggy создана Codepip • GitHub • Twitter • Settings

Want to learn CSS grid? Play Grid Garden.



FLEXBOX FROGGY

Уровень 17 из 24

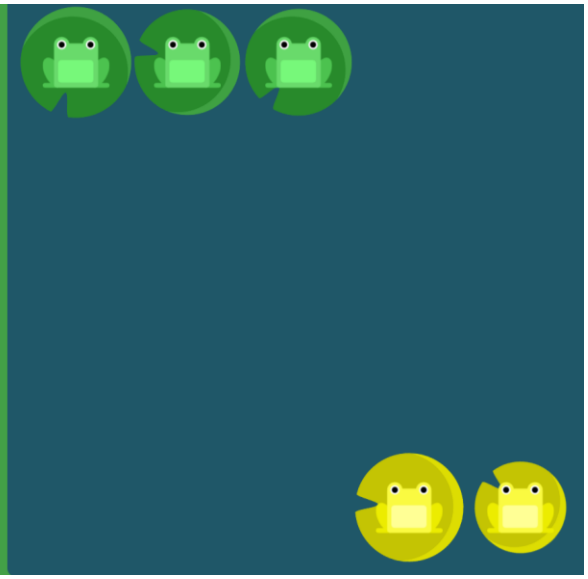
Используй `order` и `align-self` вместе, чтоб помочь лягушатам добраться к своим целям.

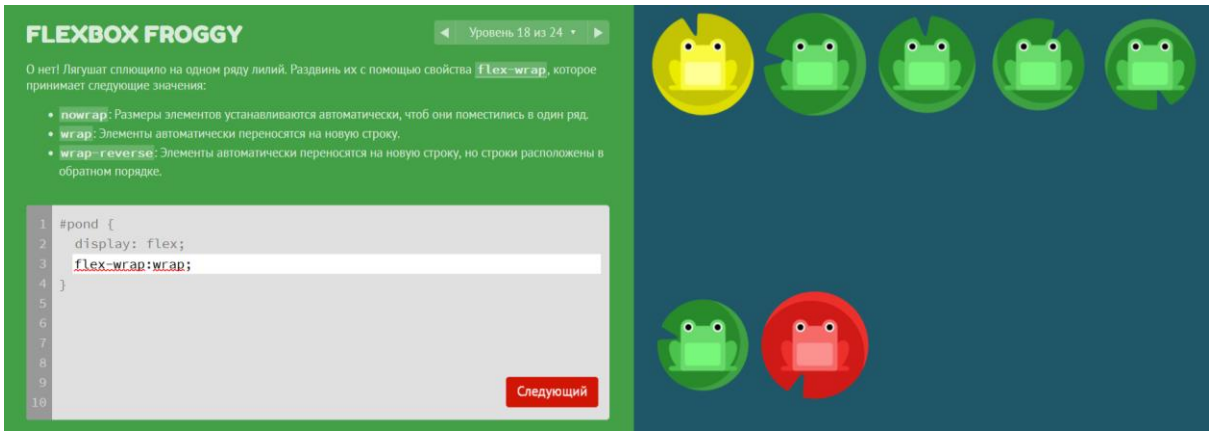
```
#pond {
  display: flex;
  align-items: flex-start;
}

.yellow {
  order: 1;
  align-self:flex-end;
}
```

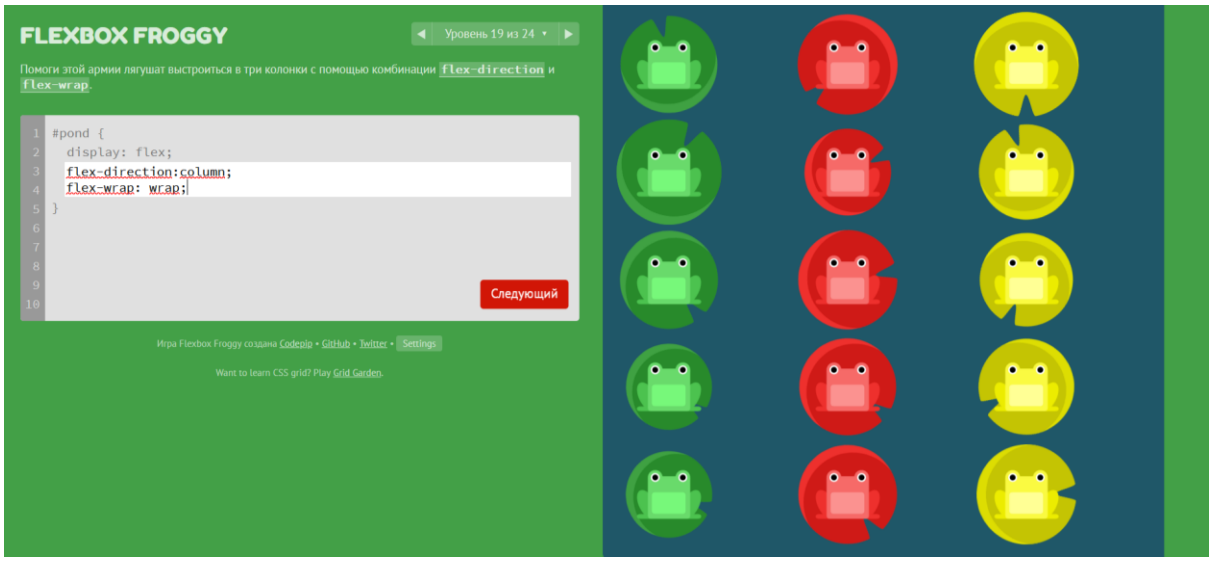
Следующий

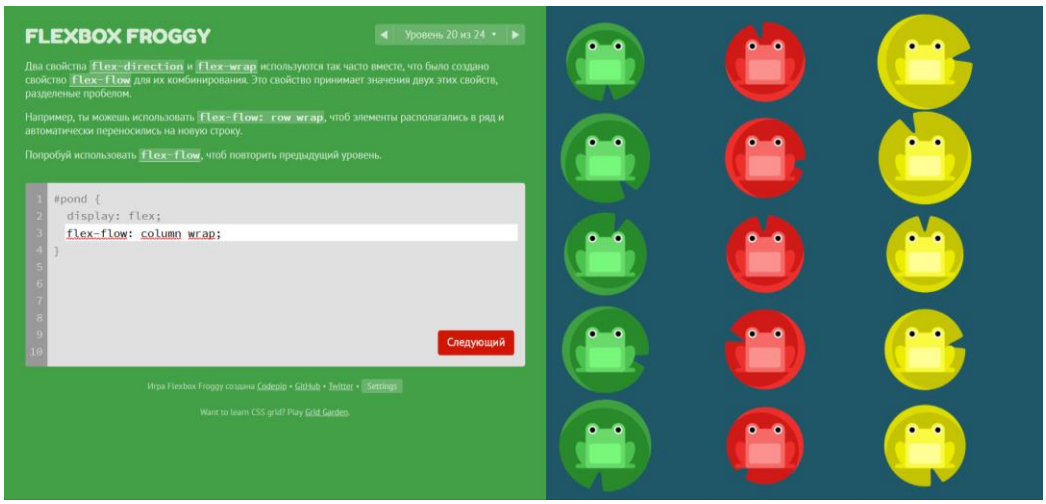
Игра Flexbox Froggy создана Codepip • GitHub • Twitter • Settings

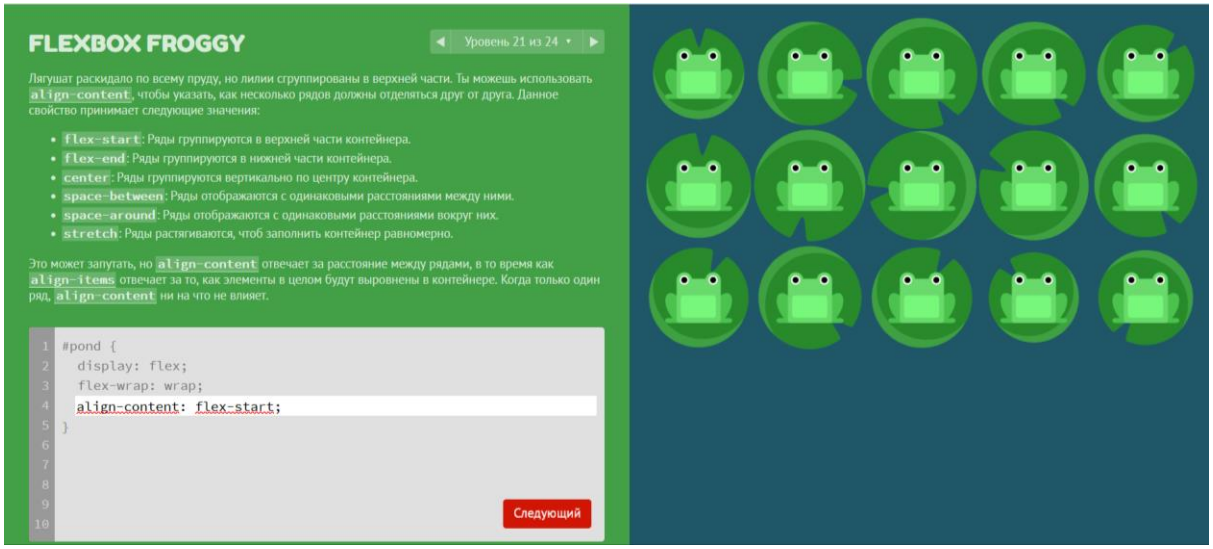
Want to learn CSS grid? Play Grid Garden.

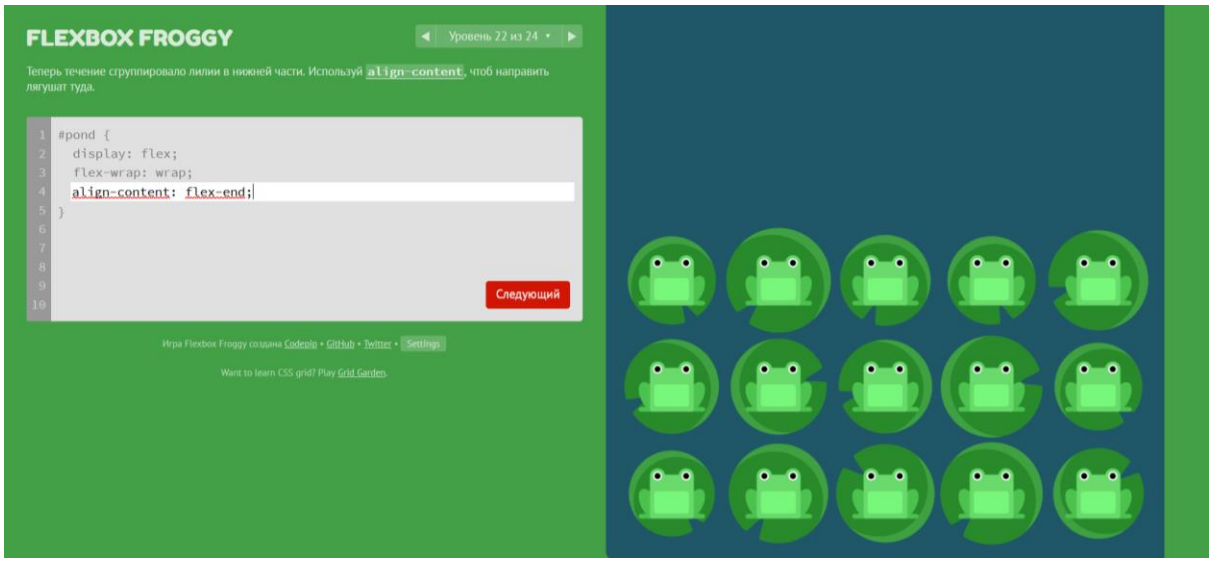


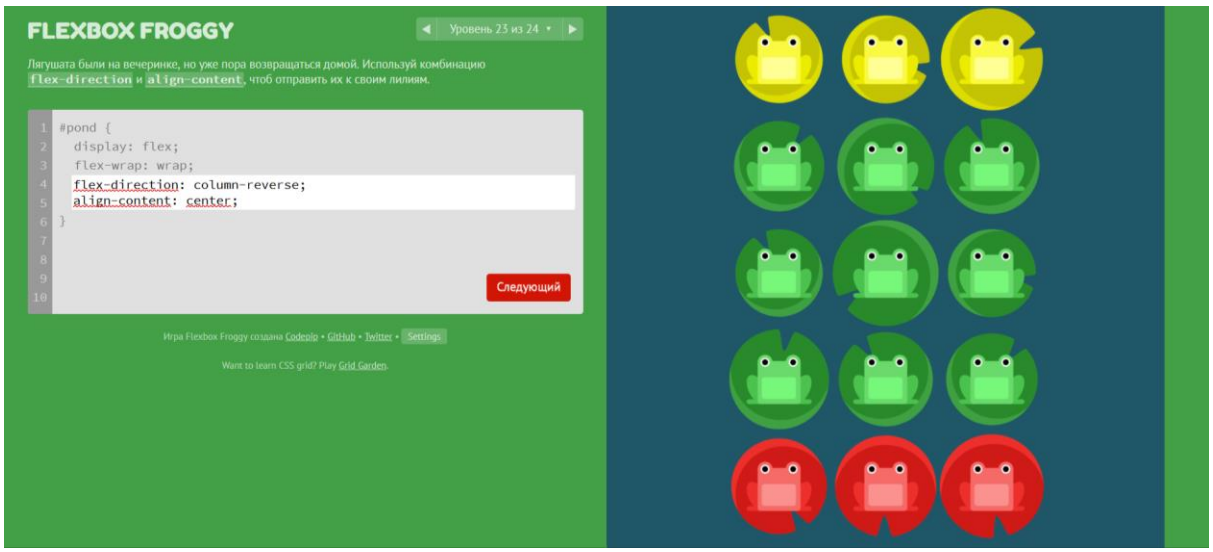

FLEXBOX FROGGY
Уровень 18 из 24
О нет! Лягушат сплющило на одном ряду лилий. Раздвинь их с помощью свойства flex-wrap, которое принимает следующие значения:
nowrap: Размеры элементов устанавливаются автоматически, чтоб они поместились в один ряд.
wrap: Элементы автоматически переносятся на новую строку.
wrap-reverse: Элементы автоматически переносятся на новую строку, но строки расположены в обратном порядке.
#pond {
display: flex;
flex-wrap:wrap;
}
Следующий

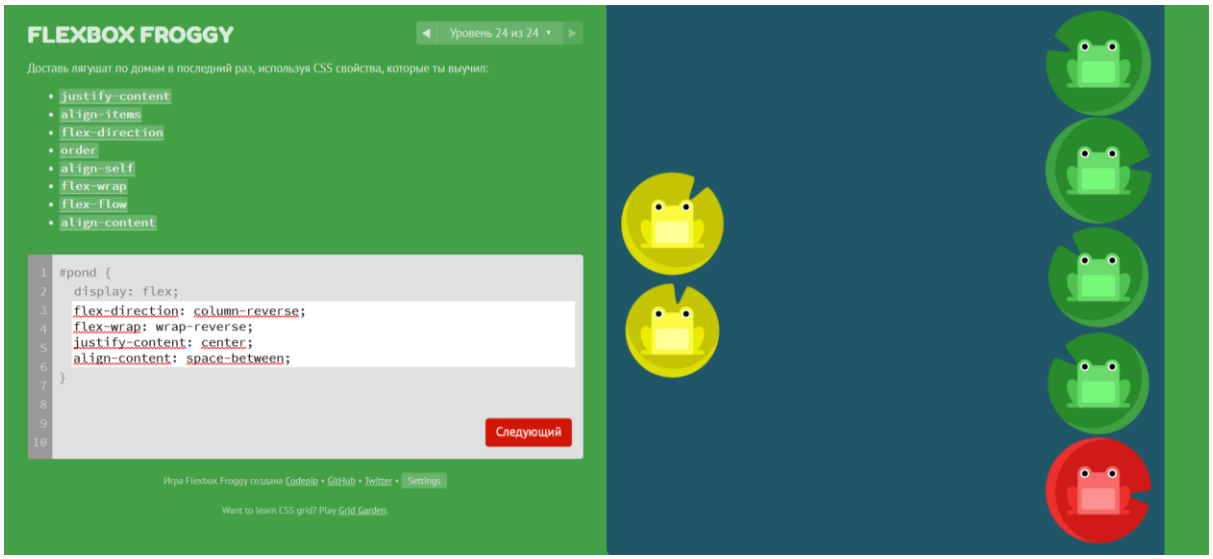

FLEXBOX FROGGY
Уровень 19 из 24
Помоги этой армии лягушат выстроиться в три колонки с помощью комбинации flex-direction и flex-wrap.
#pond {
display: flex;
flex-direction:column;
flex-wrap: wrap;
}
Следующий
Игра Flexbox Froggy создана Codepip • GitHub • Twitter • Settings
Want to learn CSS grid? Play Grid Garden.

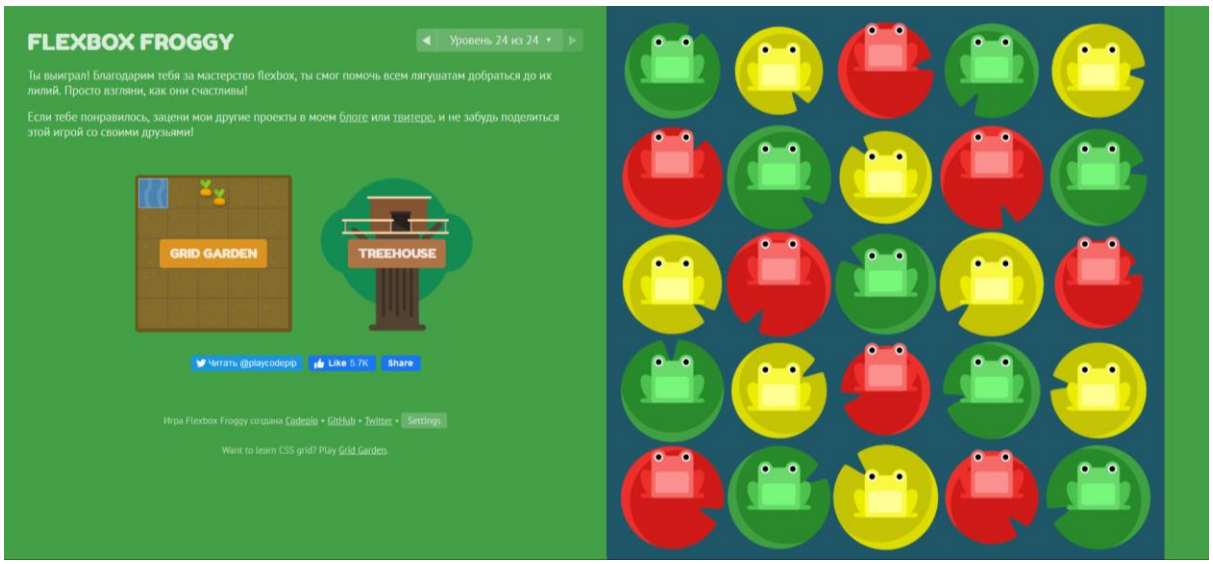

FLEXBOX FROGGY
Уровень 20 из 24
Два свойства flex-direction и flex-wrap используются так часто вместе, что было создано свойство flex-flow для их комбинирования. Это свойство принимает значения двух этих свойств, разделенные пробелом.
Например, ты можешь использовать flex-flow: row wrap, чтоб элементы располагались в ряд и автоматически переносились на новую строку.
Попробуй использовать flex-flow, чтоб повторить предыдущий уровень.
#pond {
display: flex;
flex-flow: column wrap;
}
Следующий
Игра Flexbox Froggy создана Codepip • GitHub • Twitter • Settings
Want to learn CSS grid? Play Grid Garden.


FLEXBOX FROGGY
Уровень 21 из 24
Лягушат раскидало по всему пруду, но лилии сгруппированы в верхней части. Ты можешь использовать align-content, чтобы указать, как несколько рядов должны отделяться друг от друга. Данное свойство принимает следующие значения:
flex-start: Ряды группируются в верхней части контейнера.
flex-end: Ряды группируются в нижней части контейнера.
center: Ряды группируются вертикально по центру контейнера.
space-between: Ряды отображаются с одинаковыми расстояниями между ними.
space-around: Ряды отображаются с одинаковыми расстояниями вокруг них.
stretch: Ряды растягиваются, чтоб заполнить контейнер равномерно.
Это может запутать, но align-content отвечает за расстояние между рядами, в то время как align-items отвечает за то, как элементы в целом будут выровнены в контейнере. Когда только один ряд, align-content ни на что не влияет.
#pond {
display: flex;
flex-wrap: wrap;
align-content: flex-start;
}
Следующий


FLEXBOX FROGGY
Уровень 22 из 24
Теперь течение сгруппировало лилии в нижней части. Используй align-content, чтоб направить лягушат туда.
#pond {
display: flex;
flex-wrap: wrap;
align-content: flex-end;
}
Следующий
Игра Flexbox Froggy создана Codepip • GitHub • Twitter • Settings
Want to learn CSS grid? Play Grid Garden.


FLEXBOX FROGGY
Уровень 23 из 24
Лягушата были на вечеринке, но уже пора возвращаться домой. Используй комбинацию flex-direction и align-content, чтоб отправить их к своим лилиям.
#pond {
display: flex;
flex-wrap: wrap;
flex-direction: column-reverse;
align-content: center;
}
Следующий
Игра Flexbox Froggy создана Codepip • GitHub • Twitter • Settings
Want to learn CSS grid? Play Grid Garden.





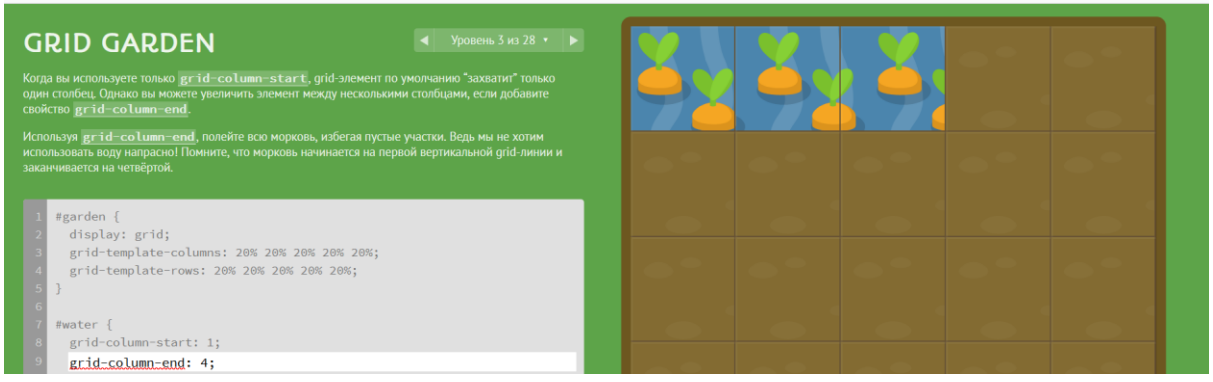
Grid Garden
Проходим постепенно все 28 уровней, познаем новые свойства.
За эти уровни мы изучили следующие свойства:

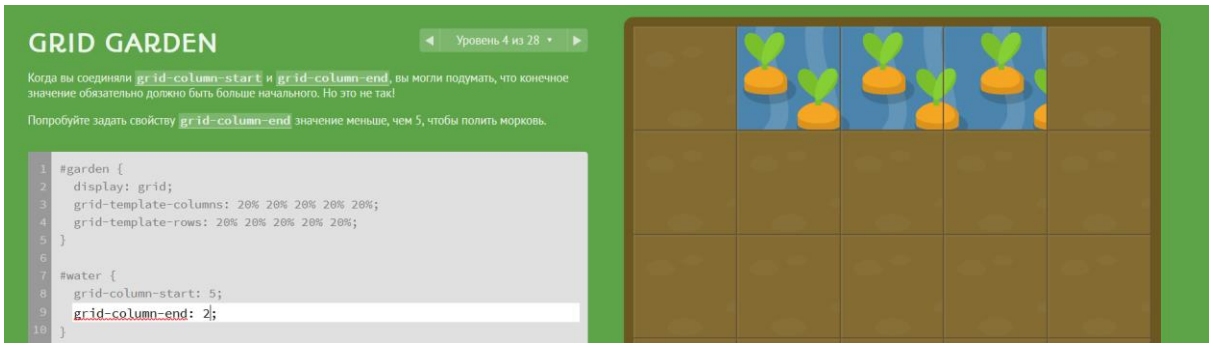
- grid-column-start (в значение можно писать положительные числа, отрицательные, span 2)
- grid-column-end
- grid-column (гибрид *grid-column-start* и *grid-column-end*)
- grid-row-start
- grid-area (принимает 4 значения, разделенные косой чертой: *grid-row-start, grid-column-start, grid-row-end* и *grid-column-end)*
- order
- grid-template-columns (можно писать 50%, repeat(8, 12.5%), 100px, 3em)
- если нужны дробные значения в grid-template-columns.

grid-template-columns: 1fr 5fr;

- grid-template — сокращённый вариант grid-template-rows и grid-template-columns

grid-template: 50% 50% / 200px





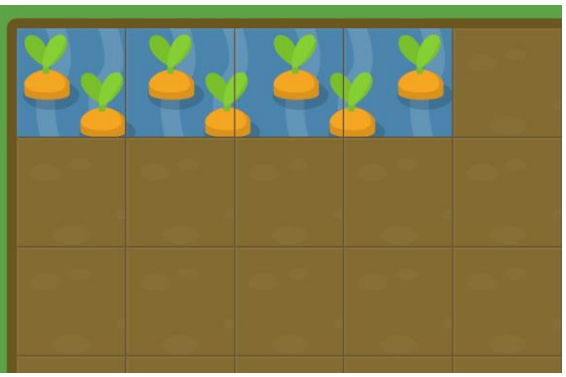
GRID GARDEN

Уровень 5 из 28

Если вы хотите посчитать grid-строки справа налево вместо слева направо, вы можете задать `grid-column-start` и `grid-column-end` отрицательные значения. Например, вы можете присвоить значение, равное `-1`, чтобы указать первую grid-строку начиная справа.

Попробуйте присвоить `grid-column-end` отрицательное значение.

```
#garden {
  display: grid;
  grid-template-columns: 20% 20% 20% 20% 20%;
  grid-template-rows: 20% 20% 20% 20% 20%;
}

#water {
  grid-column-start: 1;
  grid-column-end: -2;
}
```



GRID GARDEN

Уровень 6 из 28

Теперь попробуйте присвоить `grid-column-start` отрицательное значение.

```
#garden {
  display: grid;
  grid-template-columns: 20% 20% 20% 20% 20%;
  grid-template-rows: 20% 20% 20% 20% 20%;
}

#poison {
  grid-column-start: -3;
}
```

GRID GARDEN

Уровень 7 из 28

Вместо определения grid-элемента на основе начальной и конечной позиции grid-строк вы можете задавать их с помощью необходимой вам ширины строк, используя `span`. Помните, что `span` работает только с положительными значениями.

Для примера, полейте эту морковь, используя свойство `grid-column-end: span 2`.

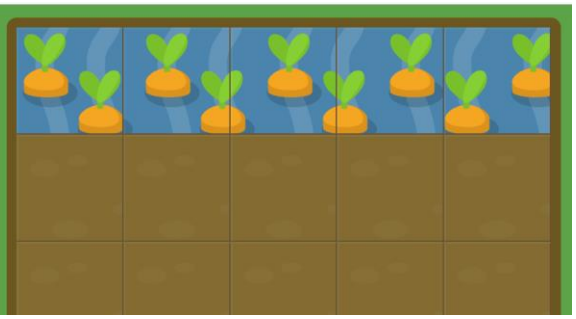
```
#garden {
  display: grid;
  grid-template-columns: 20% 20% 20% 20% 20%;
  grid-template-rows: 20% 20% 20% 20% 20%;
}

#water {
  grid-column-start: 2;
  grid-column-end: span 2;
}
```

GRID GARDEN

Уровень 8 из 28

Попробуйте опять использовать `grid-column-end` вместе со `span`, чтобы полить морковь.

```
#garden {
  display: grid;
  grid-template-columns: 20% 20% 20% 20% 20%;
  grid-template-rows: 20% 20% 20% 20% 20%;
}

#water {
  grid-column-start: 1;
  grid-column-end: span 5;
}
```



GRID GARDEN

Уровень 9 из 28

Вы также можете использовать `span` вместе с `grid-column-start`, чтобы присвоить значение ширины grid-элемента относительно конечной позиции.

```
#garden {
  display: grid;
  grid-template-columns: 20% 20% 20% 20% 20%;
  grid-template-rows: 20% 20% 20% 20% 20%;
}

#water {
  grid-column-start: span 3;
  grid-column-end: 6;
}
```

GRID GARDEN

Уровень 10 из 28

Печатать каждый раз `grid-column-start` и `grid-column-end` может быть немного утомительно. К счастью, есть краткая форма `grid-column`, которая принимает оба значения сразу через косую черту: `/`.

Например, `grid-column: 2 / 4` скажет grid-элементу начаться на второй вертикальной grid-линии и закончиться на четвёртой.

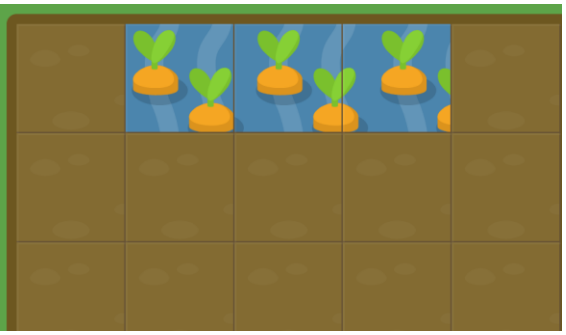
```
#garden {
  display: grid;
  grid-template-columns: 20% 20% 20% 20% 20%;
  grid-template-rows: 20% 20% 20% 20% 20%;
}

#water {
  grid-column: 4/6;
}
```

GRID GARDEN

Уровень 11 из 28

Попробуйте использовать `grid-column`, чтобы полить эту морковь. `span` также работает с этим сокращённым свойством, так что пробуйте!

```
#garden {
  display: grid;
  grid-template-columns: 20% 20% 20% 20% 20%;
  grid-template-rows: 20% 20% 20% 20% 20%;
}

#water {
  grid-column: 2/span 3
}
```



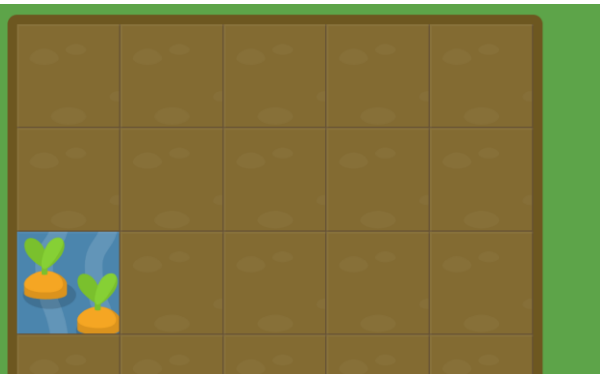
GRID GARDEN

Уровень 12 из 28

Одна из вещей, которая отличает CSS Grid Layout от Flexbox'а, – это то, что вы можете легко позиционировать элементы в двух направлениях: в столбцах и в строках. `grid-row-start` работает почти как `grid-column-start`, но только по вертикальной оси.

Используйте `grid-row-start`, чтобы полить эту морковь.

```
#garden {
  display: grid;
  grid-template-columns: 20% 20% 20% 20% 20%;
  grid-template-rows: 20% 20% 20% 20% 20%;
}

#water {
  grid-row-start: 3;
}
```



GRID GARDEN

Уровень 13 из 28

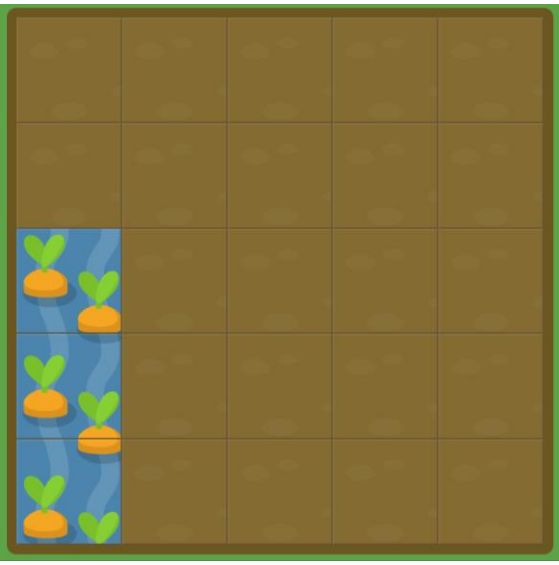
А теперь попробуйте использовать сокращённое свойство `grid-row`.

```
#garden {
  display: grid;
  grid-template-columns: 20% 20% 20% 20% 20%;
  grid-template-rows: 20% 20% 20% 20% 20%;
}

#water {
  grid-row: 3/span3;
}
```

Следующий

Игра Grid Garden создана Codepip • GitHub • Twitter • Русский

Want to learn CSS flexbox? Play Flexbox Froggy



GRID GARDEN

Уровень 14 из 28

Используйте `grid-column` и `grid-row` одновременно, чтобы разместить элемент в двух направлениях.

```
#garden {
  display: grid;
  grid-template-columns: 20% 20% 20% 20% 20%;
  grid-template-rows: 20% 20% 20% 20% 20%;
}

#poison {
  grid-column: 2;
  grid-row: 5
}
```

Следующий

Игра Grid Garden создана Codepip • GitHub • Twitter • Русский

Want to learn CSS flexbox? Play Flexbox Froggy.

GRID GARDEN

Уровень 15 из 28

Вы также можете использовать `grid-column` и `grid-row` вместе, чтобы охватить более крупные зоны внутри grid-сетки. Пробуйте!

```
#garden {
  display: grid;
  grid-template-columns: 20% 20% 20% 20% 20%;
  grid-template-rows: 20% 20% 20% 20% 20%;
}

#water {
  grid-column: 2 / span 4;
  grid-row: 1 / span 5;
}
```

Следующий

Игра Grid Garden создана Codepip • GitHub • Twitter • Русский

Want to learn CSS flexbox? Play Flexbox Froggy.

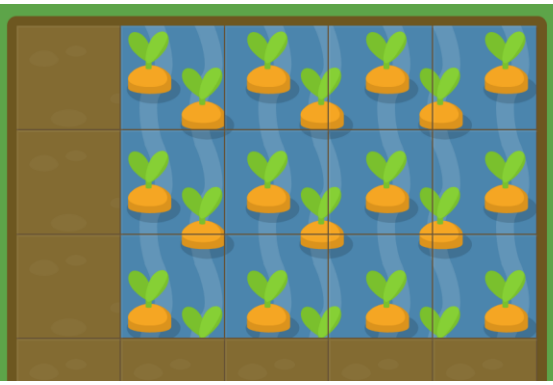
GRID GARDEN

Уровень 16 из 28

Если вас также утомляет печатать `grid-column` и `grid-row`, есть сокращённое свойство и для этого. `grid-area` принимает 4 значения, разделенные косой чертой `/`: `grid-row-start`, `grid-column-start`, `grid-row-end` и `grid-column-end`.

Пример: `grid-area: 1 / 1 / 3 / 6;`.

```
#garden {
  display: grid;
  grid-template-columns: 20% 20% 20% 20% 20%;
  grid-template-rows: 20% 20% 20% 20% 20%;
}

#water {
  grid-area: 1/2/4/6;
}
```



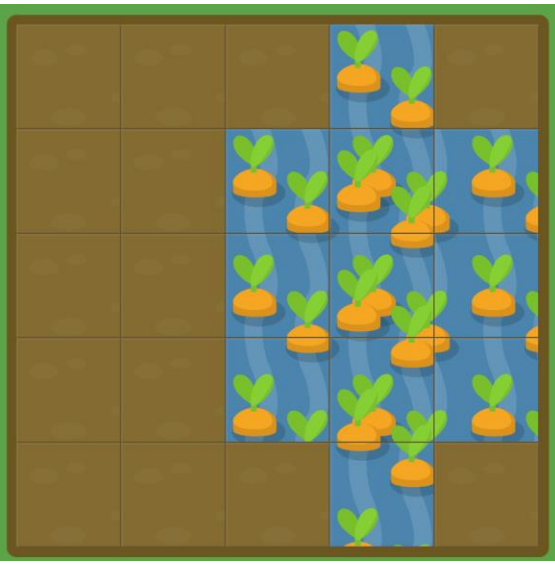
GRID GARDEN

Уровень 17 из 28

Как насчет множества элементов? Вы можете накладывать их друг на друга без проблем. Используйте `grid-area`, чтобы определить вторую зону, которая покроет всю не политую морковь.

```
#garden {
  display: grid;
  grid-template-columns: 20% 20% 20% 20% 20%;
  grid-template-rows: 20% 20% 20% 20% 20%;
}

#water-1 {
  grid-area: 1 / 4 / 6 / 5;
}

#water-2 {
  grid-area: 2/3/5/span 3;
}
```

Следующий



Игра Grid Garden создана Codepip • GitHub • Twitter • Русский

Want to learn CSS flexbox? Play Flexbox Froggy.

GRID GARDEN

Уровень 18 из 28

Если grid-элементы не имеют конкретного расположения с `grid-area`, `grid-column`, `grid-row` и т.д., они автоматически размещаются, следуя порядку, написанному в коде. Мы можем изменить это с помощью свойства `order`, которое является одним из преимуществ CSS Grid Layout перед табличной разметкой.

Изначально все grid-элементы имеют `order`, равный `0`, но этому свойству можно присвоить любое положительное или отрицательное значение, примерно как у `z-index`.

Сейчас морковь во втором столбце отравлена и сорняки в последнем столбце поливаются. Измените значение свойства `order` для яда, чтобы исправить это прямо сейчас!

```
#garden {
  display: grid;
  grid-template-columns: 20% 20% 20% 20% 20%;
  grid-template-rows: 20% 20% 20% 20% 20%;
}

.water {
  order: 0;
}

#poison {
  order: 1;
}
```

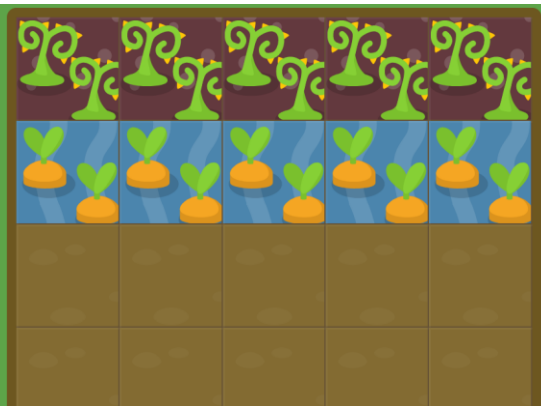
GRID GARDEN

Уровень 19 из 28

Теперь вода и яд меняются местами, хотя все сорняки находятся в начале сада. Присвойте правильное значение `order` для яда, чтобы исправить это.

```
#garden {
  display: grid;
  grid-template-columns: 20% 20% 20% 20% 20%;
  grid-template-rows: 20% 20% 20% 20% 20%;
}

.water {
  order: 0;
}

.poison {
  order: -1;
}
```

Следующий



GRID GARDEN

Уровень 20 из 28

До этого момента сад имел grid-сетку с пятью столбцами по 20% ширины и пятью строками, каждая по 20% высоты.

Это было сделано свойствами `grid-template-columns: 20% 20% 20% 20% 20%;` и `grid-template-rows: 20% 20% 20% 20% 20%;` Каждое свойство имеет пять значений, которые создают пять столбцов, где ширина каждого равна 20% от общей ширины сада.

Но вы можете изменить grid-сетку как вам вздумается. Присвойте `grid-template-columns` новое значение, чтобы полить морковь. Вам необходимо поставить значение ширины первого столбца равным 50%.

```
#garden {
  display: grid;
  grid-template-columns: 50%;
  grid-template-rows: 20% 20% 20% 20% 20%;
}

#water {
  grid-column: 1;
  grid-row: 1;
}
```

GRID GARDEN

Уровень 21 из 28

Определять несколько столбцов с одинаковой шириной может быть утомительно. К счастью, есть функция `repeat`, которая призвана помочь в этом.

Например, раньше мы определяли пять столбцов по 20% ширины с помощью `grid-template-columns: 20% 20% 20% 20% 20%;` Это можно упростить до `grid-template-columns: repeat(5, 20%)`.

Используя `grid-template-columns` вместе с функцией `repeat`, создайте 8 столбцов по 12.5% ширины каждый. Таким образом, вы не будете использовать больше воды, чем нужно.

```
#garden {
  display: grid;
  grid-template-columns: repeat(8, 12.5%);
  grid-template-rows: 20% 20% 20% 20% 20%;
}

#water {
  grid-column: 1;
  grid-row: 1;
}
```

GRID GARDEN

Уровень 22 из 28

`grid-template-columns` принимает значения не только в процентах, но и в единицах длины, например `px` или `em`. Вы даже можете комбинировать разные единицы измерения.

Присвойте трём столбцам значение `100px`, `3em` и `40%` соответственно.

```
#garden {
  display: grid;
  grid-template-columns: 100px 3em 40%;
  grid-template-rows: 20% 20% 20% 20% 20%;
}
```

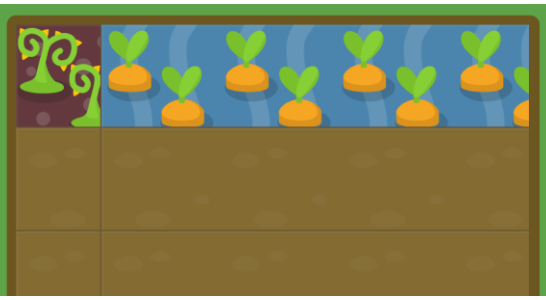
GRID GARDEN

Уровень 23 из 28

CSS Grid Layout также вводит новую единицу измерения: дробный `fr` (англ. *fraction*). Каждый `fr` выделяет одну часть свободного пространства. Например, если двум элементам задано `1fr` и `3fr` соответственно, то пространство будет поделено на 4 одинаковые части. Первый элемент займет ¼, а второй оставшиеся ¾ пространства.

Сейчас сорняки занимают левую ¼ часть вашей первой строки, а морковь оставшиеся ¾. Создайте два столбца с такой же шириной, используя `fr`.

```
#garden {
  display: grid;
  grid-template-columns: 1fr 5fr;
  grid-template-rows: 20% 20% 20% 20% 20%;
}
```



GRID GARDEN

Уровень 24 из 28

Когда одни столбцы определены с помощью пикселей, процентов или em, а любые другие столбцы — с помощью `fr`, то последние просто разделят всё возможное оставшееся пространство.

Сейчас морковь занимает 50 пикселей слева, а сорняки 50 пикселей справа. С помощью `grid-template-columns` создайте два столбца и используйте `fr`, чтобы сделать ещё 3 столбца, которые займут оставшееся простанство между ними.

```
#garden {
  display: grid;
  grid-template-columns: 50px repeat(3, 1fr) 50px;
  grid-template-rows: 20% 20% 20% 20% 20%;
}

#water {
  grid-area: 1 / 1 / 6 / 2;
}

#poison {
  grid-area: 1 / 5 / 6 / 6;
}
```

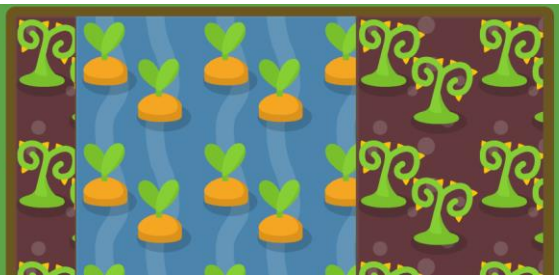
GRID GARDEN

Уровень 25 из 28

Теперь у нас есть столбец сорняков с шириной 75 пикселей на левой стороне сада. ⅗ оставшегося пространства занимает растущая морковь, а ⅖ заполонили сорняки.

Используйте `grid-template-columns` с комбинацией `px` и `fr`, чтобы создать необходимые столбцы.

```
#garden {
  display: grid;
  grid-template-columns: 75px 3fr 2fr;
  grid-template-rows: 100%;
}
```



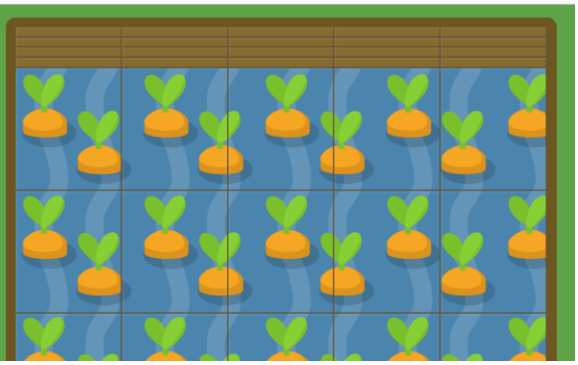
GRID GARDEN

Уровень 26 из 28

`grid-template-rows` работает точно так же, как и `grid-template-columns`.

Используйте `grid-template-rows`, чтобы полить всё, кроме верхних 50 пикселей сада. Помните, что вода заполняет только пятую строку, так что вам нужно создать именно пять строк.

```
#garden {
  display: grid;
  grid-template-columns: 20% 20% 20% 20% 20%;
  grid-template-rows: repeat(4, 12.5px);
}

#water {
  grid-column: 1 / 6;
  grid-row: 5 / 6;
}
```



GRID GARDEN

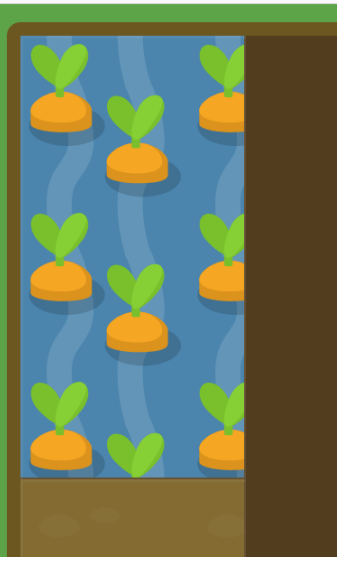
Уровень 27 из 28

`grid-template` — сокращённый вариант комбинации `grid-template-rows` и `grid-template-columns`.

Например, `grid-template: 50% 50% / 200px` создаст grid-сетку с двумя строками по 50% каждая и одним столбцом шириной 200 пикселей.

Попробуйте использовать `grid-template`, чтобы полить зону, включающую в себя верхние 60% и левые 200 пикселей сада.

```
#garden {
  display: grid;
  grid-template: 60% 40%/ 200px;
}

#water {
  grid-column: 1;
  grid-row: 1;
}
```



GRID GARDEN

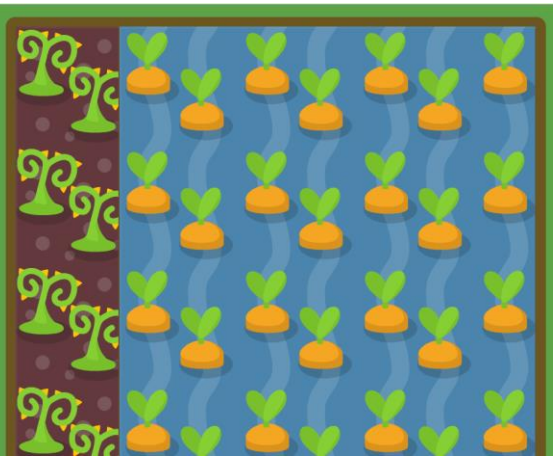
Уровень 28 из 28

Сад выглядит прекрасно. Здесь вы оставили 50-пиксельную дорожку снизу и заполнили всё оставшееся пространство морковью.

К сожалению, левые 20% сада заполонили сорняки. Используйте CSS Grid Layout в последний раз, чтобы очистить сад.

```
#garden {
  display: grid;
  grid-template: 1fr 50px / 20% 1fr;
}
```

Следующий



GRID GARDEN

Уровень 28 из 28

Вы победили! Благодаря силе CSS Grid Layout вы смогли вырастить достаточно моркови для Froggy, чтобы испечь его знаменитый 20-морковный пирог. Что, ожидали другого прыгучего друга?

Если вам понравился Grid Garden, посмотрите Flexbox Froggy, чтобы изучить другие новые возможности CSS. Вы также можете следить за моими проектами в моём блоге или в твиттере.

Хотите поддержать Grid Garden? Зацените топовые курсы веб-дизайна и программирования от Treehouse. И расскажите своим друзьям и семье про Grid Garden!

Вывод

Во время выполнения практической работы 1 было пройдено 2 игры по изучению свойств CSS Flexbox и CSS Grid.