

**САНКТ-ПЕТЕРБУРГСКИЙ НАЦИОНАЛЬНЫЙ
ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО**

Дисциплина: Фронт-энд разработка

Отчет

Домашняя работа №1

Выполнил:

Сидельникова Дарья

Группа J3211

**Проверил:
Добряков Д. И.**

**Санкт-Петербург
2026 г.**

Задача

Пройти 3 обучающие игры

Ход работы

1-ая игра «FLEXBOX FROGGY»

Что выучила во время выполнения:

(Вся теория с прохождения игры)

justify-content:

- flex-start: элементы выравниваются по левой стороне контейнера.
- flex-end: элементы выравниваются по правой стороне контейнера.
- center: элементы выравниваются по центру контейнера.
- space-between: отображаются с одинаковыми отступами между ними.
- space-around: отображаются с одинаковыми отступами вокруг них.

align-items:

- 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, чтобы заполнить контейнер.

flex-direction:

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

flex-wrap

- nowrap: размеры элементов устанавливаются автоматически, чтобы 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.

align-content:

- 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: отображаются с одинаковыми расстояниями между ними.
- space-around: ряды отображаются с одинаковыми расстояниями вокруг них.
- stretch: ряды растягиваются, чтобы заполнить контейнер равномерно.

Когда в качестве направления выбрана колонка, то **justify-content** влияет на вертикальное выравнивание, а **align-items** — на горизонтальное.

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

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.

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

Два свойства **flex-direction** и **flex-wrap** используются так часто вместе, что было создано свойство **flex-flow** для их комбинирования. Это свойство принимает их значения, разделённые пробелом. (**flex-flow: row wrap**)

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.

FLEXBOX FROGGY

Уровень 24 из 24

Ты выиграл! Благодарим тебя за мастерство flexbox, ты смог помочь всем лягушатам добраться до своих лилий. Просто взгляни, как они счастливы!

Если тебе понравилось, зацени мои другие проекты в моем блоге или твиттере, и не забудь поделиться этой игрой со своими друзьями!

Grid Garden

Anchoreum

Disarray

CSS Scoops

Игра Flexbox Froggy создана [Codepen](#) • [YouTube](#) • [Twitter](#) • [GitHub](#) • [Настройки](#)

Хотите изучить CSS grid? Сыграйте в [Grid Garden](#) or [Anchoreum](#).

FLEXBOX FROGGY

Уровень 24 из 24

Доставь лягушат домой в последний раз, используя свойства CSS, которые ты выучил:

- [justify-content](#)
- [align-items](#)
- [flex-direction](#)
- [order](#)
- [align-self](#)
- [flex-wrap](#)
- [flex-flow](#)
- [align-content](#)

```

1 #pond {
2   display: flex;
3   flex-flow: column-reverse wrap-reverse;
4   justify-content: center;
5   align-content: space-between;
6 }
7
8
9
10

```

Следующий

Игра Flexbox Froggy создана [Codepen](#) • [YouTube](#) • [Twitter](#) • [GitHub](#) • [Настройки](#)

Хотите изучить CSS grid? Сыграйте в [Grid Garden](#) or [Anchoreum](#).

2-ая игра «GRID GARDEN»

Что выучила во время выполнения:

(Вся теория с прохождения игры)

- grid-column-start
- grid-column-end
- span - ширина размещения (работает только с положительными значениями)
- grid-column: 2 / 4 (+ span)
- grid-row-start
- grid-row(также как grid-column: 2 / 4 (+ span))
- grid-area принимает 4 значения, разделенные косой чертой /: grid-row-start, grid-column-start, grid-row-end и grid-column-end (grid-area: 1 / 1 / 3 / 6;)
- 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.
- grid-template-columns: 20% 20% 20% 20% 20%; и grid-template-rows: 20% 20% 20% 20%; Каждое свойство имеет пять значений, которые создают пять столбцов, где ширина каждого равна 20% от общей ширины сада (grid-template-columns: repeat(5, 20%))
- 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. 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` `3fr` соответственно, то пространство будет поделено на 4 одинаковые части. Первый элемент займет $\frac{1}{4}$, а второй оставшиеся $\frac{3}{4}$ пространства.

- `grid-template-rows` работает точно так же, как и `grid-template-columns`.
- `grid-template` — сокращённый вариант комбинации `grid-template-rows` и `grid-template-columns`

GRID GARDEN

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 пространство между ними.

```

1 #garden {
2   display: grid;
3   grid-template-columns: 50px 1fr 1fr 1fr 50px;
4   grid-template-rows: 20% 20% 20% 20% 20%;
5 }
6
7 #water {
8   grid-area: 1 / 1 / 6 / 2;
9 }
10
11 #poison {
12   grid-area: 1 / 5 / 6 / 6;
13 }
14

```

[Следующий](#)

Игра Grid Garden создана [Codepen](#) • [YouTube](#) • [Twitter](#) • [GitHub](#) • [Русский](#)

Want to learn more CSS? Play [Flexbox Froggy](#) or [Anchoreum](#).

GRID GARDEN

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 [моем блоге](#) или в [твиттере](#).

И расскажите своим друзьям и семье про Grid Garden!

Flexbox Froggy

Anchoreum

Disarray

CSS Scoops

Игра Grid Garden создана [Codepen](#) • [YouTube](#) • [Twitter](#) • [GitHub](#) • [Русский](#)

Want to learn more CSS? Play [Flexbox Froggy](#) or [Anchoreum](#).

GRID GARDEN

Уровень 28 из 28 ▶

Сад выглядит прекрасно. Здесь вы оставили 50-пиксельную дорожку снизу и заполнили всё оставшееся пространство морковью. Используйте CSS Grid Layout в последний раз, чтобы очистить сад.

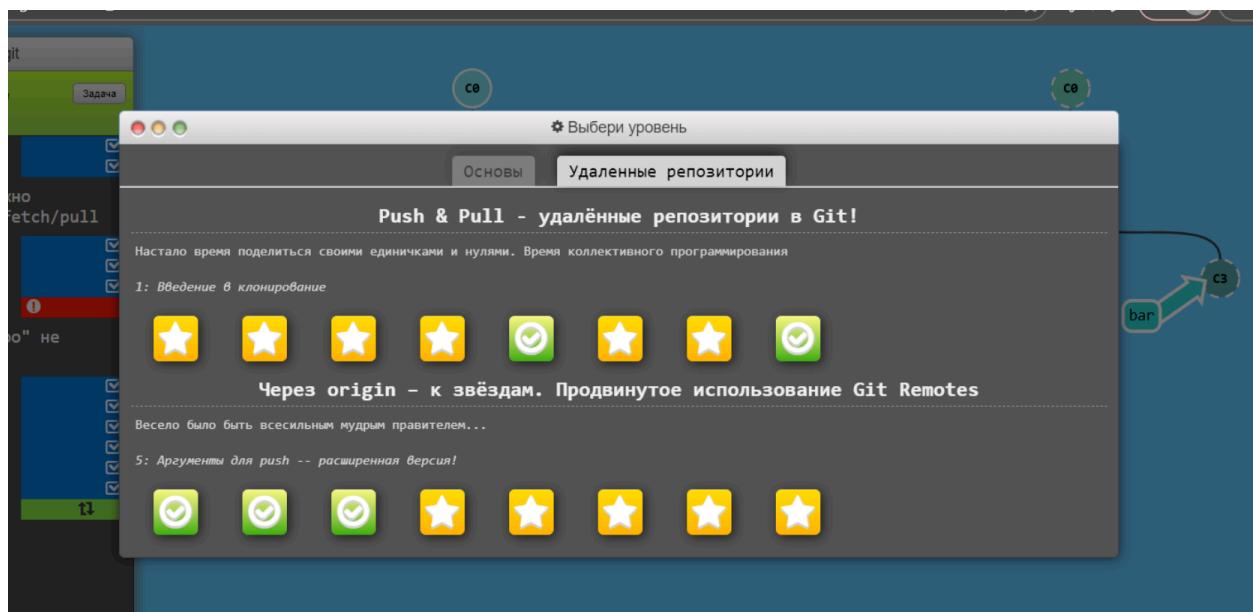
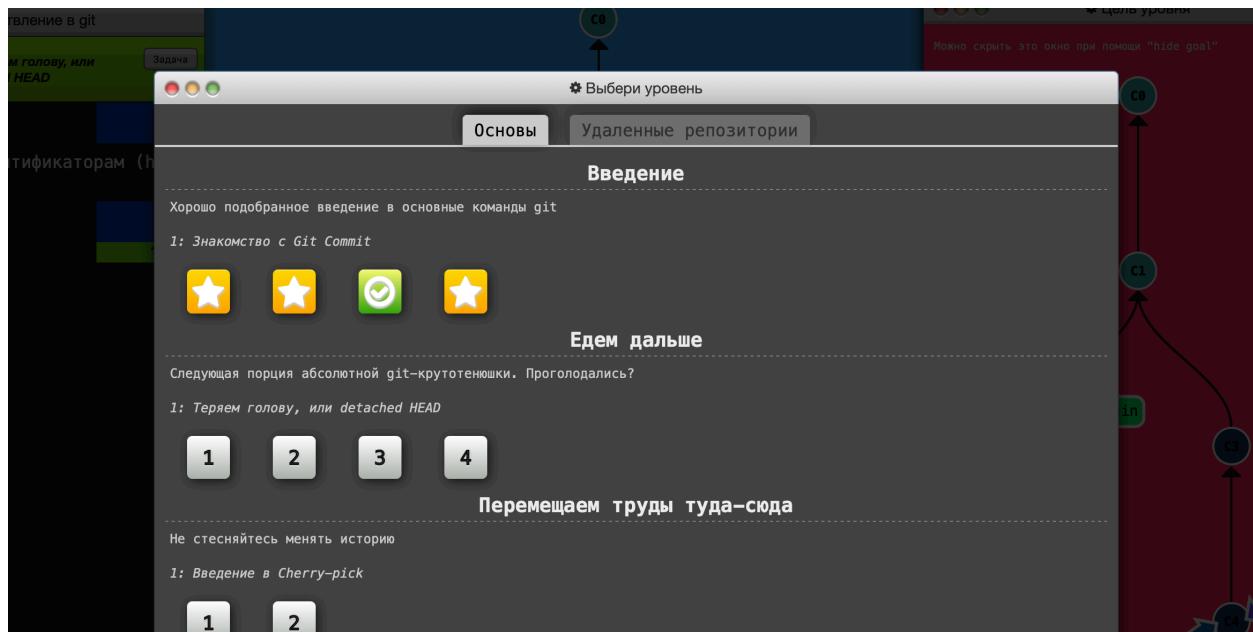
```
1 #garden {  
2   display: grid;  
3   grid-template: 1fr 50px / 20% 80%;  
4 }  
5  
6  
7  
8  
9  
10  
11  
12  
13  
14
```

Следующий

Игра Grid Garden создана [CodePen](#) • [YouTube](#) • [Twitter](#) • [GitHub](#) • [Русский](#)

Want to learn more CSS? Play [Flexbox Froggy](#) or [Anchoreum](#).

3-я игра «Learn Git Branching»



Выход

Во время прохождения игр, я изучила grid и flexbox из css, а также разобралась с git