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.

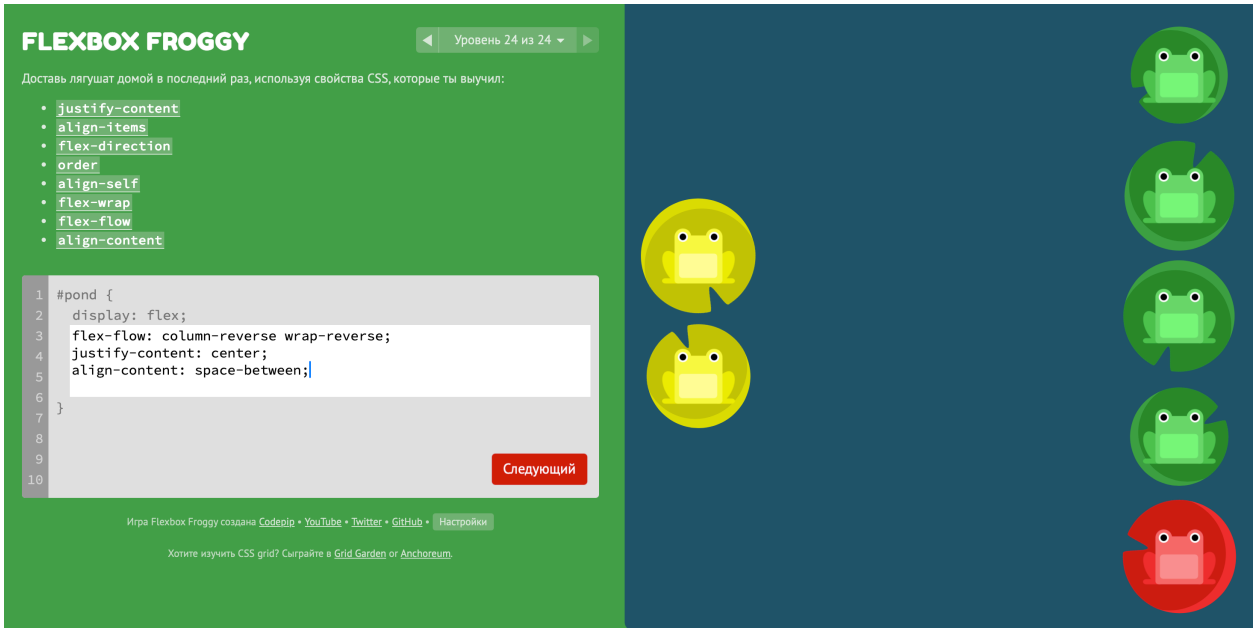
Когда ты устанавливаешь направление в обратном порядке для ряда или колонки, начало (start) и конец (end) тоже меняются местами.

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.



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.

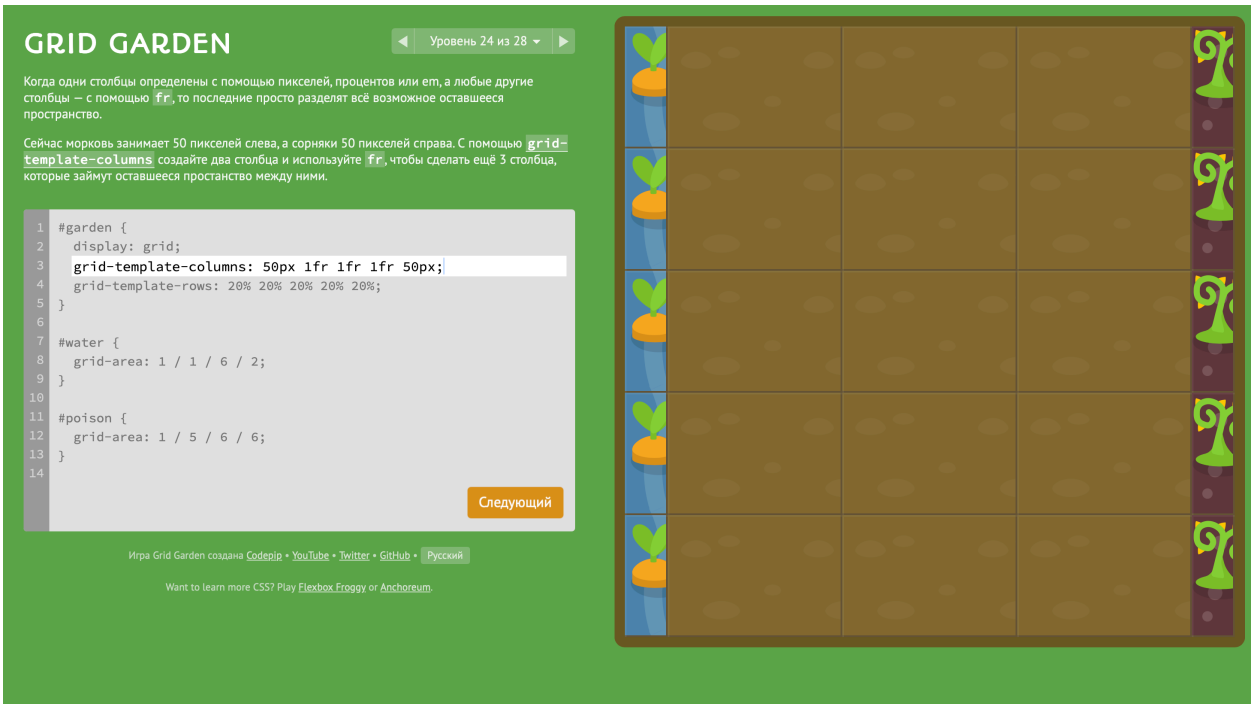
- grid-template-columns: 20% 20% 20% 20% 20%; и grid-template-rows: 20% 20% 20% 20% 20%; Каждое свойство имеет пять значений, которые создают пять столбцов, где ширина каждого равна 20% от общей ширины сада (grid-template-columns: repeat(5, 20%))

- 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

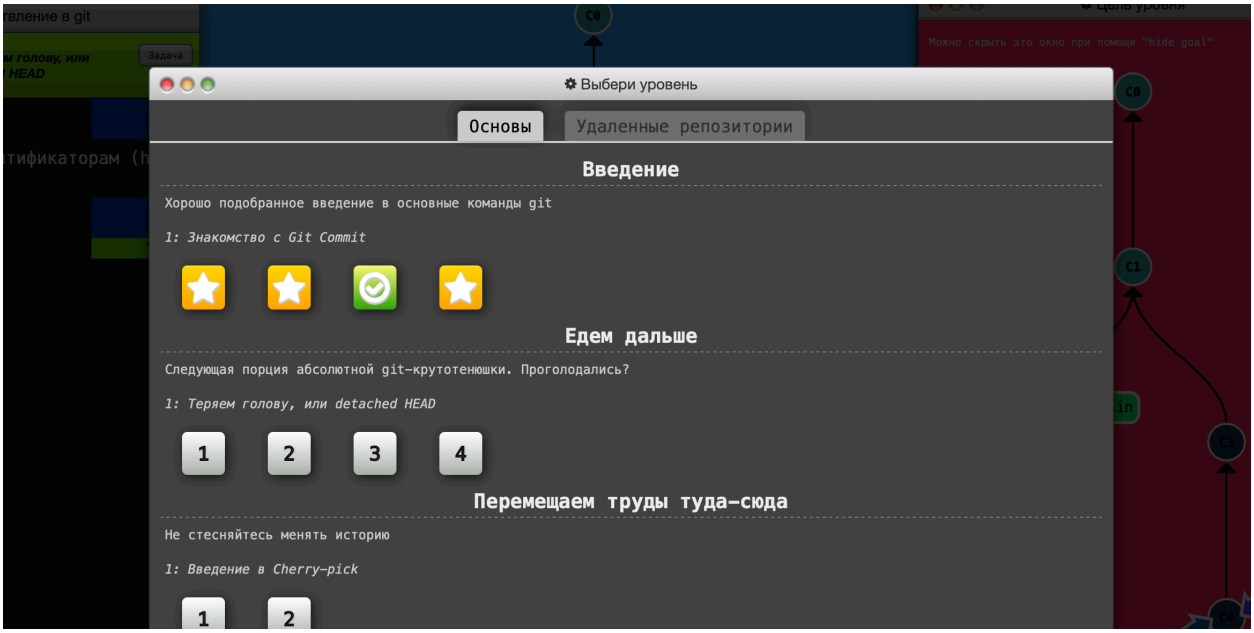
свободного пространства. Например, если двум элементам задано 1fru 3fr соответственно, то пространство будет поделено на 4 одинаковые части. Первый элемент займет ¼, а второй оставшиеся ¾ пространства.

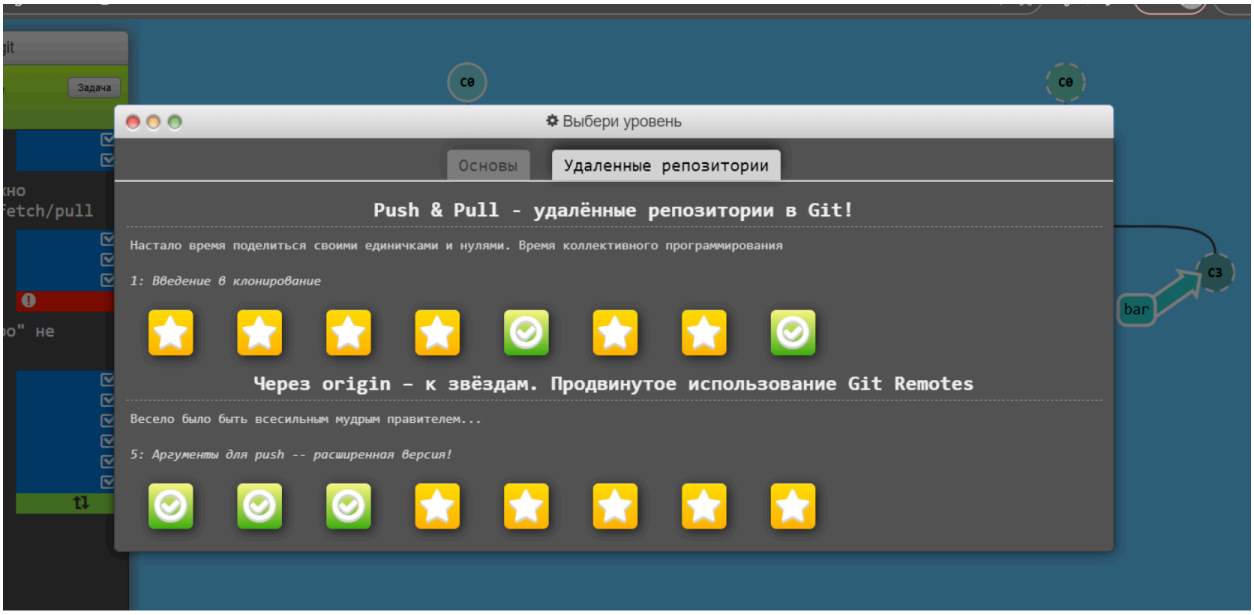
- grid-template-rows работает точно так же, как и grid-template-columns.

- grid-template — сокращённый вариант комбинации grid-template-rows и grid-template-column



3-я игра «Lear Git Branching»





Вывод

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