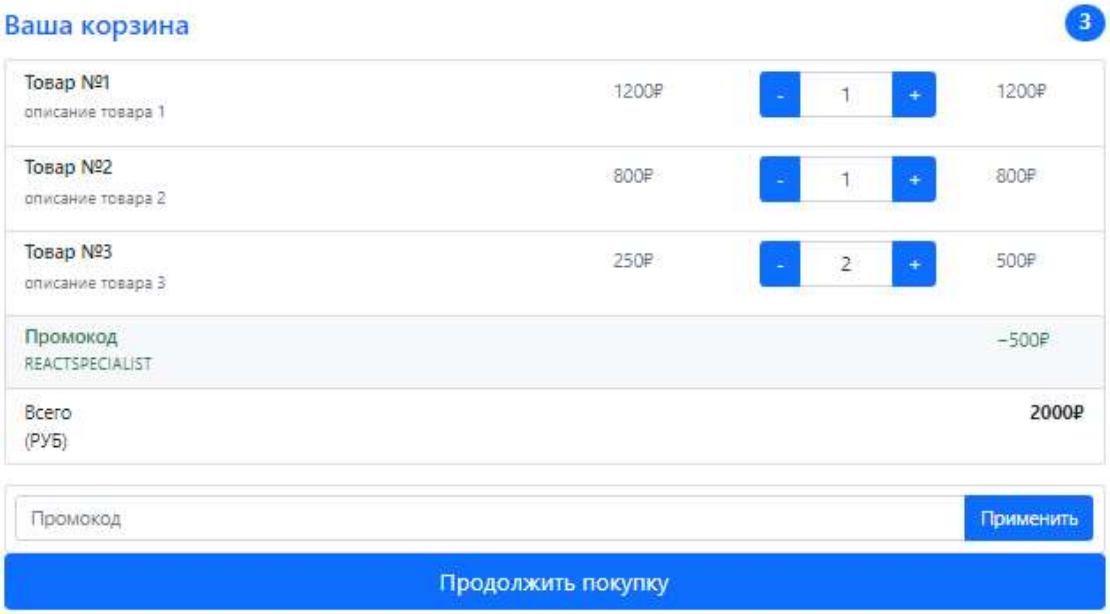
Лабораторная работа: Композиция компонентов в React

В этой лабораторной мы попрактикуемся создавать композицию компонентов для создания некоторой корзины на сайте. **Если вы не выполняли предыдущие лабораторные**, создайте новую папку `03mod-my-app` и откройте её в консоли. Установите в ней CRA. **Примечание**: основная папка проекта в VSCode по-прежнему должна называться *my_app*



Упражнение 1: Создание композиции

1.1 Подумайте, как блоки на изображении которые могут стать компонентами. Подумайте (но не создавайте) над названиями будущих компонентов.

Например, набор компонентов может быть такой:

- `Basket.js` - основной компонент отображающий корзину. Принимает список добавленных в корзину товаров, а также их количество и цену
- `BasketItem.js` - отображает отдельный товар в корзине
- `BasketPromoInfo.js` - отображает информацию о применённом промокоде
- `BasketPromoCode.js` - позволяет вводить промокод
- `BasketTotal.js` - указывает суммарную стоимость заказа с учетом вычета по промокоду
- `ItemInfo.js` - название и описание отдельного товара
- `Counter.js` - счетчики количества товара для отдельного товара
- `BasketHeader.js` - верхняя часть корзины
- `Button.js` - отдельная кнопка
- `ItemPrice.js` - информация о цене

1.2. Познакомьтесь с рекомендациями по структуре React-проектов. В папке **components** создайте файлы компонентов и папку **css**:

```
src
  ├── components
  |     ├── css
  |     ├── Basket.js
  |     ├── Button.js
  |     ├── BasketItem.js
  |     ├── BasketPromoInfo.js
  |     ├── BasketPromoCode.js
  |     ├── BasketTotal.js
  |     ├── Button.js
  |     ├── ItemInfo.js
  |     ├── ItemPrice.js
  |     ├── Counter.js
  |     ├── BasketHeader.js
  |     └── Button.css
  ├── App.css
  ├── App.js
...
```

1.3 Создайте для каждого компонента шаблонную заготовку, типа такой

```
import React from 'react'

const ComponentName = () => {
  return (
    <div>ComponentName</div>
  )
}

export default ComponentName
```

- `import React from 'react'` - импорт React
- `const ComponentName = () => {}` - описание функционального компонента как функции-стрелки
- `export default ComponentName` - экспорт созданного компонента

Примечание: можно установить VSCode-плагин es7-react-js-snippets. Он позволяет использовать сниппеты, то есть сокращения , которые превращаются в нужный фрагмент кода. Например, при вызове `rafce+Tab` внутри файла **BasketHeader.js** будет создана заготовка

```
import React from 'react'

const BasketHeader = () => {
  return (
    <div>BasketHeader</div>
  )
}

export default BasketHeader
```

1.4. Добавьте в компонент `App` константу `items` с массивом товаров и укажите использование компонента `Basket`.

```
import "./App.css";
import Basket from "./components/Basket.js";

function App() {
  const items = [
    {
      uid: "86ed58db-082d-45ab-aa81-5218059349cb",
      title: "Товар1",
      description: "описание товара 1",
      price: 1200,
      qty: 1,
    },
    {
      uid: "05542e59-7a90-4e80-bf9d-78967f272049",
      title: "Товар2",
      description: "описание товара 2",
      price: 800,
      qty: 1,
    },
    {
      uid: "7793e4f0-fe86-47cc-98f6-e01b6beeb3af",
      title: "Товар3",
      description: "описание товара 3",
      price: 250,
      qty: 2,
    },
  ];

  return (
    <div className="App">
      <header className="App-header">
        <h1>Корзина</h1>

        <Basket items={items} />
      </header>
    </div>
  );
}

export default App;
```

1.5. Убедитесь, что при запущенном через `npm start` приложении, страница приложения http://localhost:3000/ запущена, работает и не содержит ошибки.

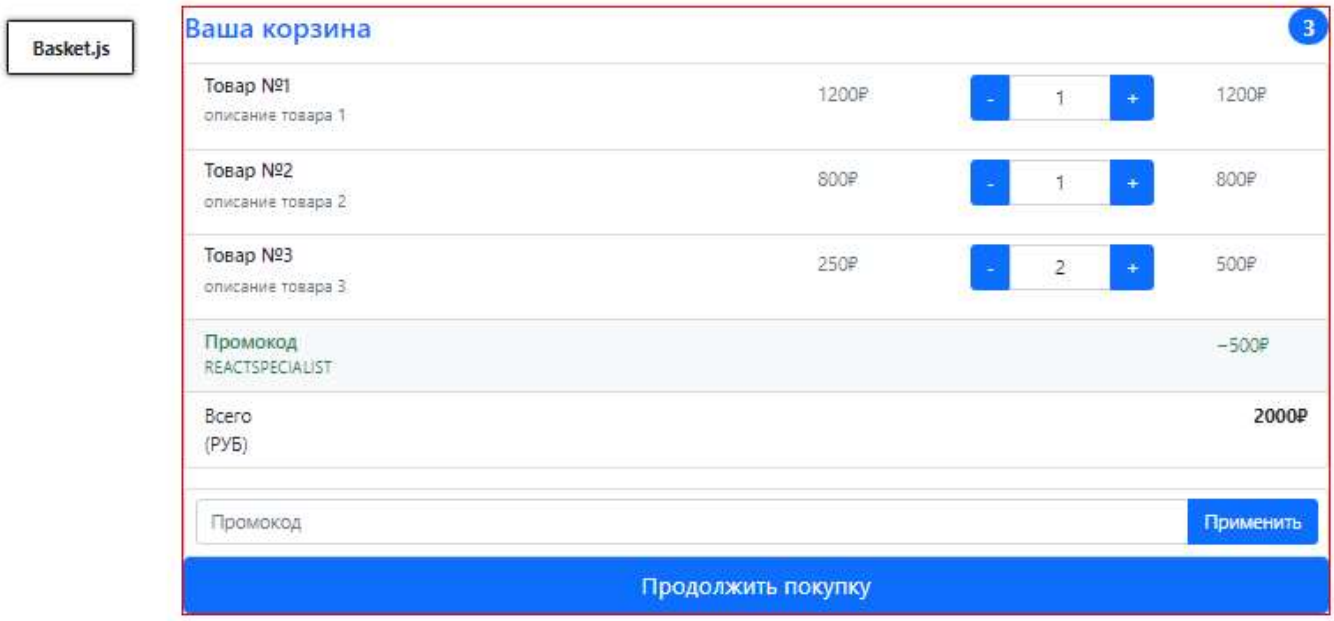
1.6. Если ошибки есть - исправьте их.

1.7. Познакомьтесь с утверждениями и убедитесь, что все их них понятны. Если что-то непонятно, спросите преподавателя.

- компонент `App` - главный компонент
- `App` представляет собой наше приложение
- `App` импортирует все файлы, которые будут им использоваться
- `import "./App.css"` тут происходит импорт стилей из файла **App.css**
- `import Basket from "./components/Basket.js"` - сейчас импортируется только `Basket`
- `<Basket items={items} />` - используем `Basket` внутри возвращаемого JSX
- в файле компонента происходит импорт нужных стилей и других JS-файлов

Упражнение 2: Создание компонента `Basket.js`

В этом упражнении нужно описать содержимое компонента `Basket`. На изображении показывается где находится компонент. При создании, подумайте какие свойства/props принимает компонент, как компонент соотносится с другими компонентами



2.1. Измените описание компонента `Basket`

```
import React from "react";
import "./css/Basket.css";

import BasketHeader from "./BasketHeader";
import BasketItem from "./BasketItem";
import BasketPromoInfo from "./BasketPromoInfo";
import BasketTotal from "./BasketTotal";
import BasketPromoCode from "./BasketPromoCode";
import Button from "./Button";

const Basket = ({ items }) => {
  const countItemsInBasket = items.reduce((acc, next) => acc + next.qty, 0);
  const amountTotal = items.reduce(
    (acc, next) => acc + next.price * next.qty,
    -500
  );

  return (
    <div className="Basket">
      <BasketHeader count={countItemsInBasket} />

      <div className="Basket__items">
        {items.map((item) => (
          <BasketItem {...item} key={item.uid} />
        ))}

        <BasketPromoInfo code={"REACTSPECIALIST"} />
        <BasketTotal value={amountTotal} currency={"₽"} />
      </div>

      <BasketPromoCode code={""} />
      <Button
        value="Продолжить покупку"
        onClickHandler={() => alert("Продолжить")}
        className="btn-proceed"
      />
    </div>
  );
};

export default Basket;
```

2.2. Создайте для него стилевое описание **my_app/components/css/Basket.css**

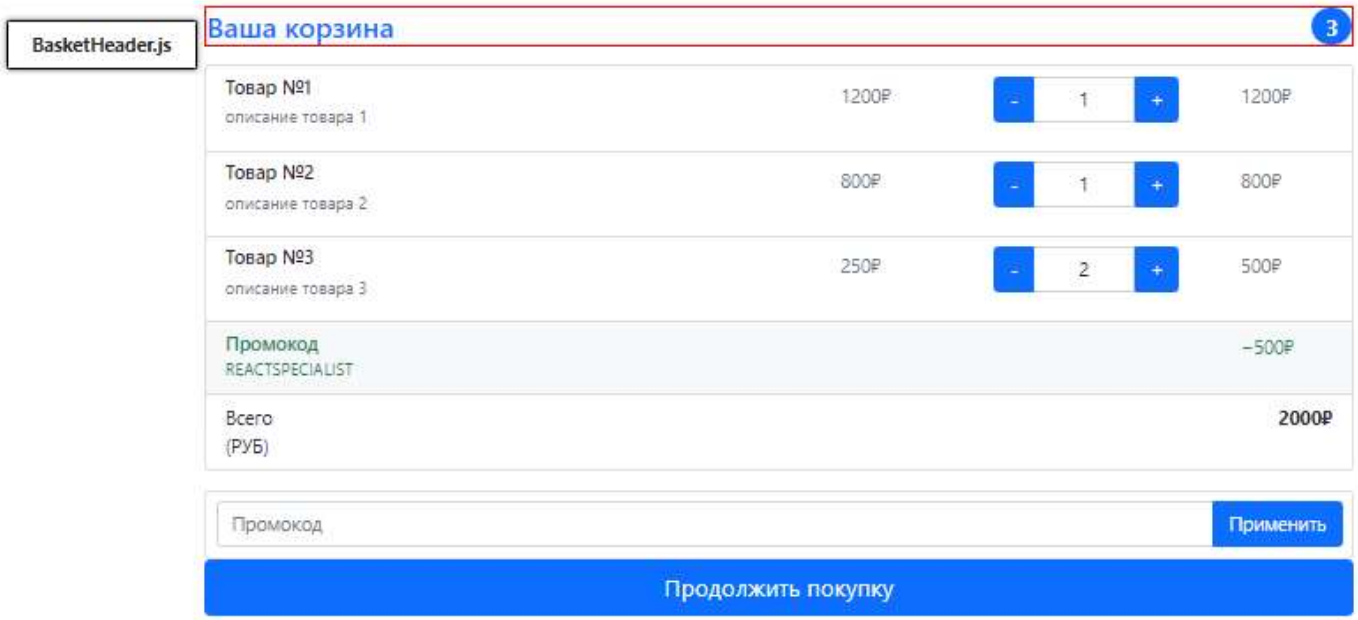
```
.Basket * {
  box-sizing: border-box;
  /* border: 1px solid #f00; */
}
.Basket {
  width: 80%;
  margin: auto;
}

.BasketItem,
.BasketPromoInfo,
.BasketTotal {
  padding: 10px 20px;
}
```

2.3. Познакомьтесь с утверждениями и убедитесь, что все их них понятны. Если что-то непонятно, спросите преподавателя.

- компонент `Basket` - отрисовывает корзину на сайте
- `Basket` будет *уметь*
  - изменять количество добавленных товаров
  - находить сумму по товарам
  - принимать промокод
- `import "./css/Basket.css"` - стили компонента `Basket` лежат в отдельном файле
- Array.prototype.reduce() - метод массивов для свёртки
- `(acc, next) => acc + next` - стрелочная функция, принимающая два аргумента `acc` и `next`, а возвращающая их сумму
- `<div className="Basket">` - для компонента задаём собственный класс (пробуем БЭМ, но без усердия)
- `{items.map((item) => ( JSX )}` - на основе массива объектов `items` через `map()` возвращаем новый массив с JSX-элементами
- `const Basket = ({ items }) => {...` - при помощи деструктуризации получаем из свойств/props свойство `items`. Это удобная альтернатива выражению `let items = props.items`

Упражнение 3: Создание компонента `BasketHeader.js`

В этом упражнении нужно описать содержимое компонента `BasketHeader`. На изображении показывается где находится компонент. При создании, подумайте какие свойства/props принимает компонент, как компонент соотносится с другими компонентами



3.1. Измените описание компонента `BasketHeader`

```
import React from 'react';
import './css/BasketHeader.css';

const BasketHeader = (props) => {
  return (
    <div className='BasketHeader'>
      <h2 className='BasketHeader_h2'>Ваша корзина</h2>
      <span className='BasketHeader_count'>{props.count}</span>
    </div>
  );
};

export default BasketHeader;
```

3.2. Создайте для него стилевое описание **my_app/components/css/BasketHeader.css**

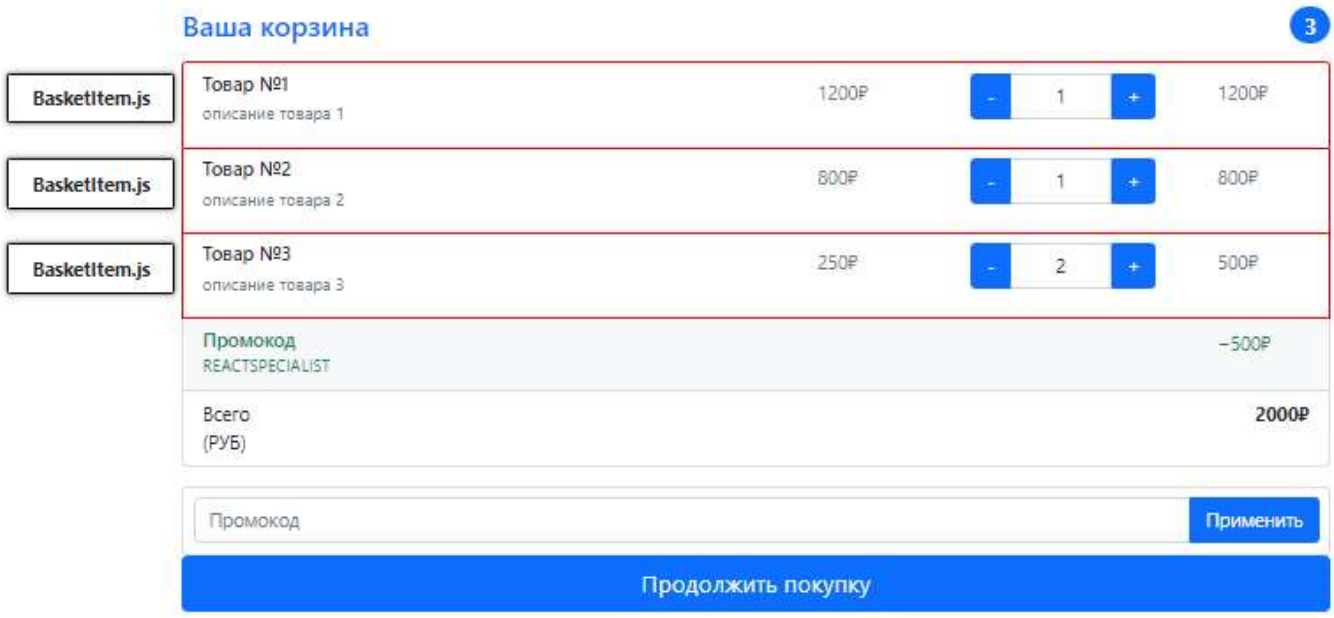
```
.BasketHeader {
  display: flex;
  margin: 10px 0;
}

.BasketHeader_h2 {
  width: 95%;
  font-size: 1em;
  color: rgb(0, 87, 250);
}
.BasketHeader_count {
  min-width: 20px;
  height: 20px;
  background-color: rgb(0, 87, 250);
  color: #fff;
  border-radius: 20px;
  text-align: center;
  line-height: 1.8em;
  font-size: 0.7em;
}
```

3.3. Познакомьтесь с утверждениями и убедитесь, что все их них понятны. Если что-то непонятно, спросите преподавателя.

- компонент `BasketHeader` - отрисовывает верхнюю часть корзины
- `BasketHeader` будет *уметь*
  - выводить количество единиц товаров (а не количество *позиций*)

Упражнение 4: Создание компонента `BasketItem.js`

В этом упражнении нужно описать содержимое компонента `BasketItem`. На изображении показывается где находится компонент. При создании, подумайте какие свойства/props принимает компонент, как компонент соотносится с другими компонентами



4.1. Измените описание компонента `BasketItem`

```
import React from 'react'
import "./css/BasketItem.css"

import ItemInfo from './ItemInfo'
import ItemPrice from './ItemPrice'
import Counter from './Counter'

const BasketItem = ({
  uid,
  title,
  description,
  price,
  qty,
}) => {

  return (
    <div className='BasketItem'>
      <ItemInfo title={title} description={description} />
      <ItemPrice value={price} currency={'₽'} />
      <Counter value={qty} uid={uid} />
      <ItemPrice value={qty * price} currency={'₽'}  />
    </div>
  )
}

export default BasketItem
```

4.2. Создайте для него стилевое описание **my_app/components/css/BasketItem.css**

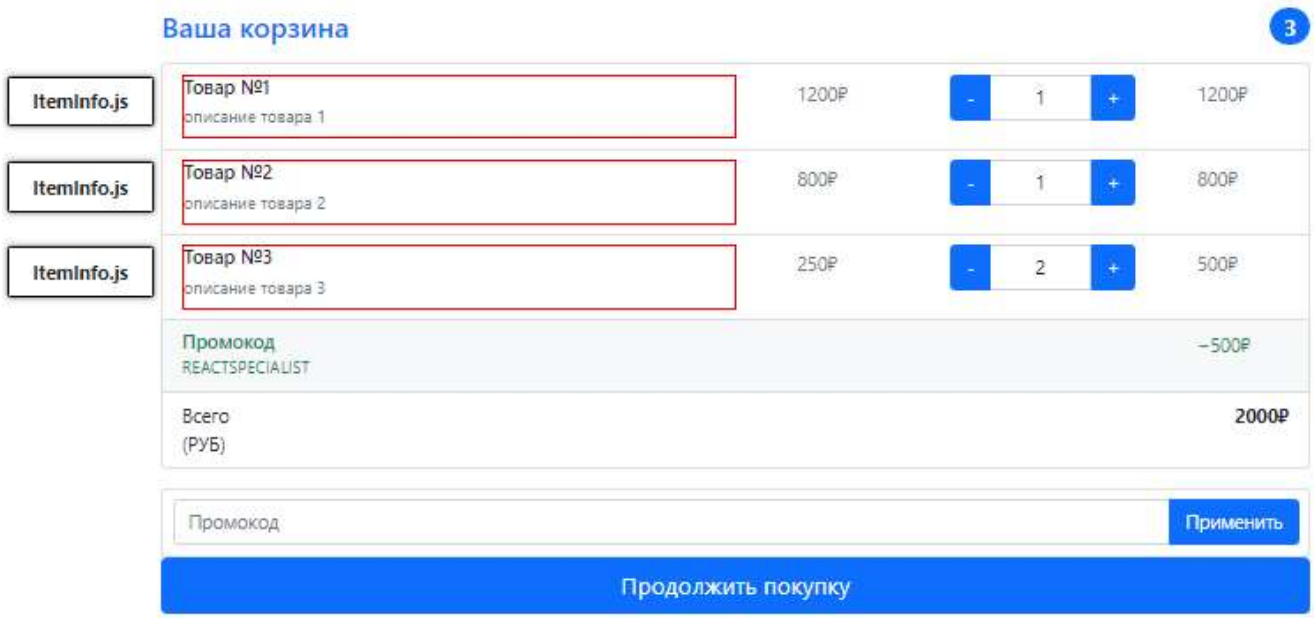
```
.BasketItem {
  display: flex;
  border: 1px solid rgb(235, 235, 235);
}
.BasketItem:hover {
  background: rgba(245, 245, 245, 0.5);
}
```

4.3. Познакомьтесь с утверждениями и убедитесь, что все их них понятны. Если что-то непонятно, спросите преподавателя.

- компонент `BasketItem` - отдельную позицию/товар, добавленный в корзину
- `BasketItem` будет *уметь*
  - выводить название и описание отдельного товара в корзине
  - выводить цену одной единицы товара в корзине
  - выводить стоимость *нужного* количества товаров
  - указанные возможности `BasketItem` реализует со вспомогательными компонентами
- `const BasketItem = ({uid, title, description})` - быстрое получение свойств/пропсов/props из первого аргумента функции

Упражнение 5: Создание компонента `ItemInfo.js`

В этом упражнении нужно описать содержимое компонента `ItemInfo`. На изображении показывается где находится компонент. При создании, подумайте какие свойства/props принимает компонент, как компонент соотносится с другими компонентами



5.1. Измените описание компонента `ItemInfo`

```
import React from 'react'
import "./css/ItemInfo.css"

const ItemInfo = ({title, description}) => {
  return (
    <div className='ItemInfo'>
      <h3 className='ItemInfo_title'>{title}</h3>
      <p className='ItemInfo_description'>{description}</p>
    </div>
  )
}

export default ItemInfo
```

5.2. Создайте для него стилевое описание **my_app/components/css/ItemInfo.css**

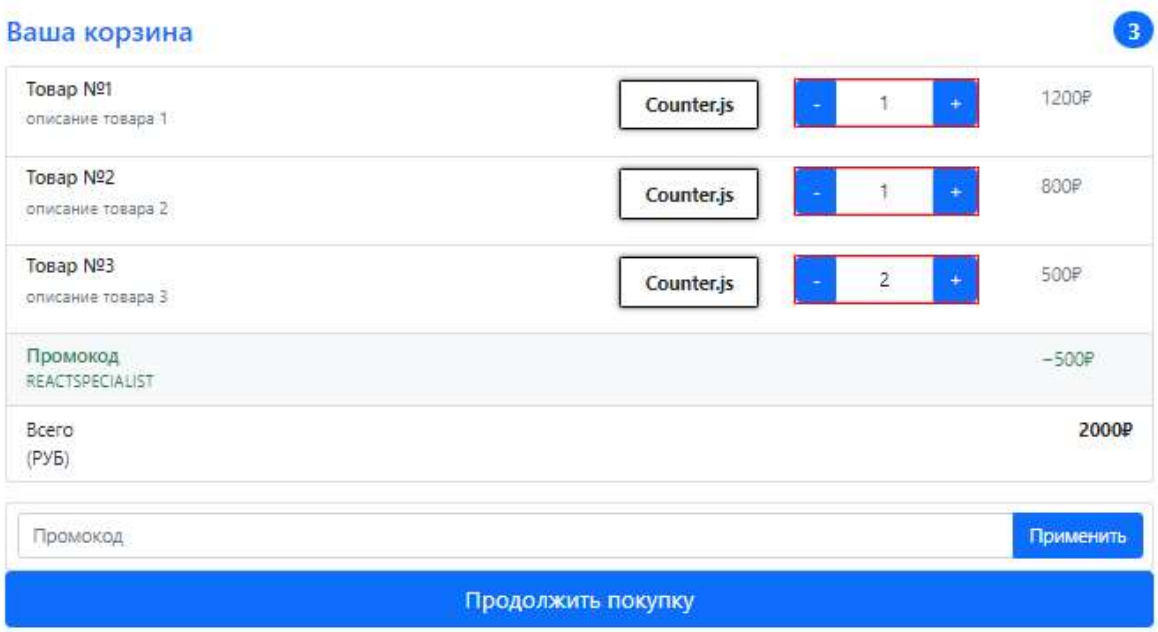
```css
.ItemInfo {
  width: 60%;
}

.ItemInfo_title {
  font-size: 1em;
}

.ItemInfo_description {
  color: rgb(145, 145, 145);
}
```

5.3. Познакомьтесь с утверждениями и убедитесь, что все их них понятны. Если что-то непонятно, спросите преподавателя.

- компонент `ItemInfo` - вспомогательный для `BasketItem` компонент
- `ItemInfo` будет *уметь*
  - выводить название отдельного товара в корзине
  - выводить описание отдельного товара в корзине
- *можно* избавиться от этого компонента, если перенести его содержимое в сам `BasketItem` (на этом курсе мы так не сделаем)

Упражнение 6: Создание компонента `Counter.js`

В этом упражнении нужно описать содержимое компонента `Counter`. На изображении показывается где находится компонент. При создании, подумайте какие свойства/props принимает компонент, как компонент соотносится с другими компонентами



6.1. Измените описание компонента `Counter`

```
import React from "react";
import "./css/Counter.css";

import Button from "./Button";

const Counter = ({ value, uid }) => {
  return (
    <div className="Counter">
      <div className="Counter_into">
        <Button
          value="-"
          onClickHandler={() => {
            alert("-" + uid);
          }}
        />
        <input
          className="Counter_input"
          defaultValue={value}
          onChange={(ev) => {
            console.log(ev.target.value);
          }}
        />
        <Button
          value="+"
          onClickHandler={() => {
            alert("+" + uid);
          }}
        />
      </div>
    </div>
  );
};

export default Counter;
```

6.2. Создайте для него стилевое описание **my_app/components/css/Counter.css**

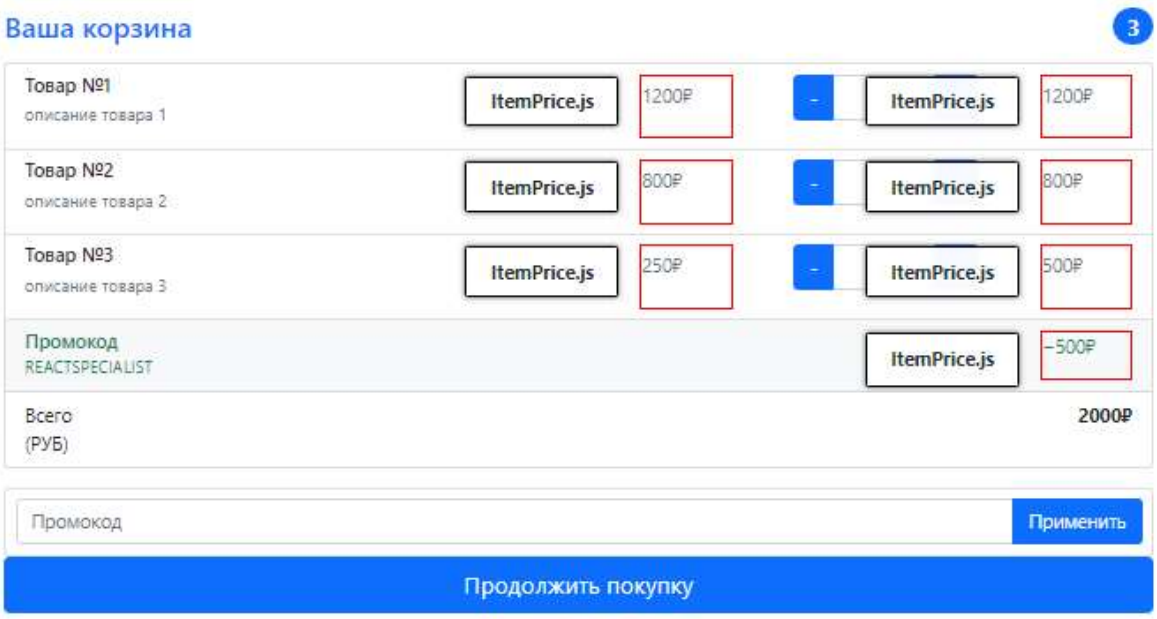
```
.Counter {
  width: 30%;
  line-height: 2em;
}
.Counter_into {
  margin: auto;
  width: 80%;
}
.Counter_input {
  width: 60px;
  text-align: center;
  padding: 5px 10px;
  border-radius: 3px;
  border: 1px solid rgb(235, 235, 235);
}
.Counter_input:focus {
  outline: 1px solid rgb(204, 204, 204);
}

@media screen and (max-width: 700px){
    .Counter {
        width: 40%;
        line-height: 2em;
    }
    .Counter_input {
        width: 40px;
    }
}
```

6.3. Познакомьтесь с утверждениями и убедитесь, что все их них понятны. Если что-то непонятно, спросите преподавателя.

- компонент `Counter` - вспомогательный для `BasketItem` компонент
- `Counter` будет *уметь*
  - выводить количество добавленных в корзину товаров
  - увеличивать или уменьшать количество товаров за счёт `Button`
  - изменять количество товара за счет изменения `<input />`

Упражнение 7: Создание компонента `ItemPrice.js`

В этом упражнении нужно описать содержимое компонента `ItemPrice`. На изображении показывается где находится компонент. При создании, подумайте какие свойства/props принимает компонент, как компонент соотносится с другими компонентами



7.1. Измените описание компонента `ItemPrice`

```
import React from 'react'
import "./css/ItemPrice.css"

const ItemPrice = ({value, currency}) => {
  return (
    <div className='ItemPrice'>{value}{currency}</div>
  )
}

export default ItemPrice
```

7.2. Создайте для него стилевое описание **my_app/components/css/ItemPrice.css**

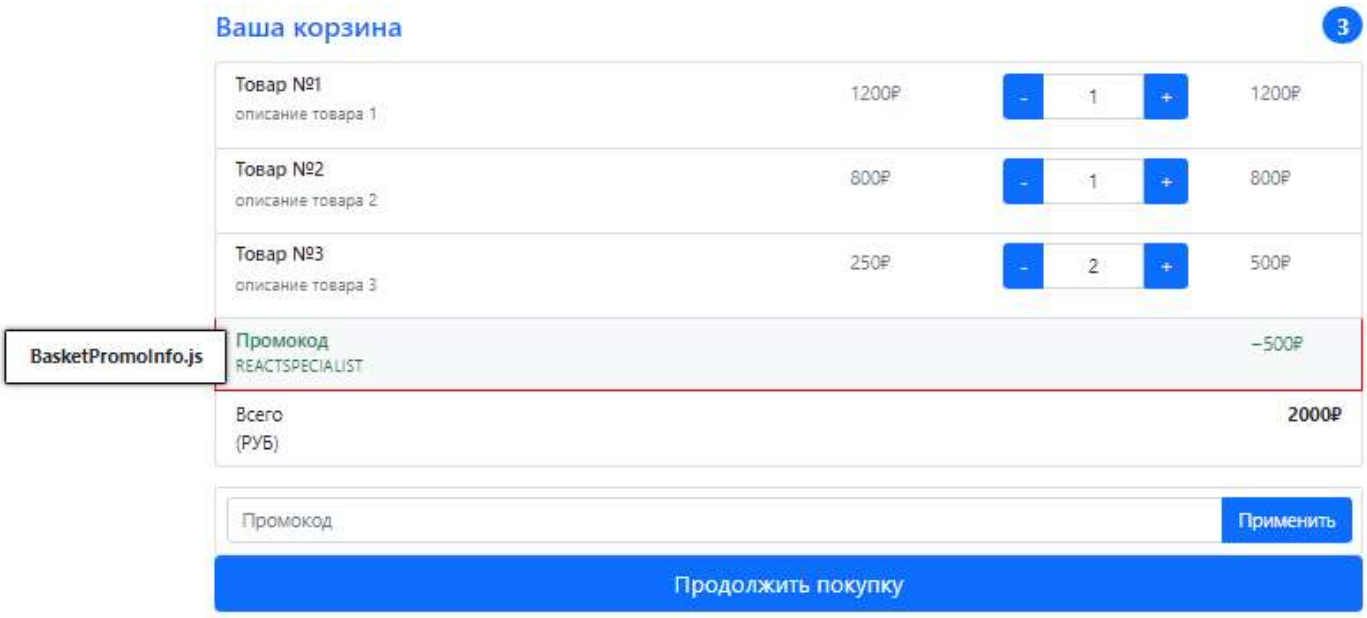
```
.ItemPrice {
  color: rgb(145, 145, 145);
  width: 10%;
  line-height: 2em;
}
```

7.3. Познакомьтесь с утверждениями и убедитесь, что все их них понятны. Если что-то непонятно, спросите преподавателя.

- компонент `ItemPrice` - вспомогательный для `BasketItem` компонент
- `ItemPrice` будет *уметь*
  - только выводить стоимость и знак используемой валюты (*передаётся через свойства*)
- это один из самых простых компонентов в работе

Упражнение 8: Создание компонента `BasketPromoInfo.js`

В этом упражнении нужно описать содержимое компонента `BasketPromoInfo`. На изображении показывается где находится компонент. При создании, подумайте какие свойства/props принимает компонент, как компонент соотносится с другими компонентами



8.1. Измените описание компонента `BasketPromoInfo`

```
import React from "react";
import "./css/BasketPromoInfo.css";

import ItemPrice from './ItemPrice'

const BasketPromoInfo = ({ code }) => {
  return (
    <div className="BasketPromoInfo">
      <div className='BasketPromoInfo_main'>
        <h3 className="BasketPromoInfo_title">Промокод</h3>
        <p className="BasketPromoInfo_description">{code}</p>
      </div>
      <div className='BasketPromoInfo_price'>
        <ItemPrice value={-500} currency={'₽'} />
      </div>
    </div>
  );
};

export default BasketPromoInfo;
```

8.2. Создайте для него стилевое описание **my_app/components/css/BasketPromoInfo.css**

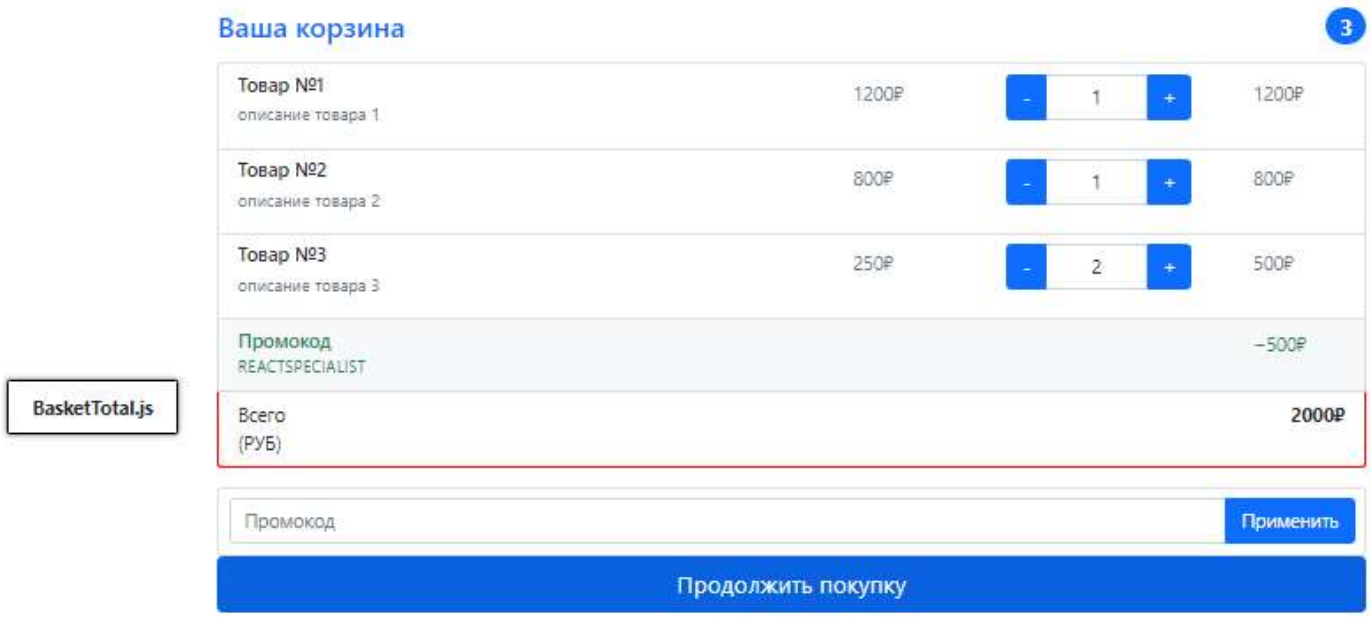
```css
.BasketPromoInfo {
  display: flex;
  background: rgb(248, 251, 245);
  color: rgb(0, 86, 0);
  border: 1px solid rgb(235, 235, 235);
}

.BasketPromoInfo_main {
  width: 92%;
}

.BasketPromoInfo_title {
  font-size: 1em;
}
.BasketPromoInfo_description {
  font-size: 0.8em;
  text-transform: uppercase;
}
```

8.3. Познакомьтесь с утверждениями и убедитесь, что все их них понятны. Если что-то непонятно, спросите преподавателя.

- компонент `BasketPromoInfo` - компонент похожий на `BasketItem` - отображает строку с введённым промокодом
- `BasketPromoInfo` будет *уметь*
  - выводить название промокода
  - выводить размер скидки (*размер скидки приходит через свойства*)

Упражнение 9: Создание компонента `BasketTotal.js`

В этом упражнении нужно описать содержимое компонента `BasketTotal`. На изображении показывается где находится компонент. При создании, подумайте какие свойства/props принимает компонент, как компонент соотносится с другими компонентами



9.1. Измените описание компонента `BasketTotal`

```
import React from "react";
import "./css/BasketTotal.css";

import ItemPrice from "./ItemPrice";

const BasketTotal = (props) => {
  const { value, currency } = props;

  return (
    <div className="BasketTotal">
      <div className="BasketTotal_main">
        <h3 className="BasketTotal_title">Всего</h3>
        <p className="BasketTotaldescription">(РУБ)</p>
      </div>
      <div className="BasketTotal_price">
        <ItemPrice value={value} currency={currency} />
      </div>
    </div>
  );
};

export default BasketTotal;
```

9.2. Создайте для него стилевое описание **my_app/components/css/BasketTotal.css**

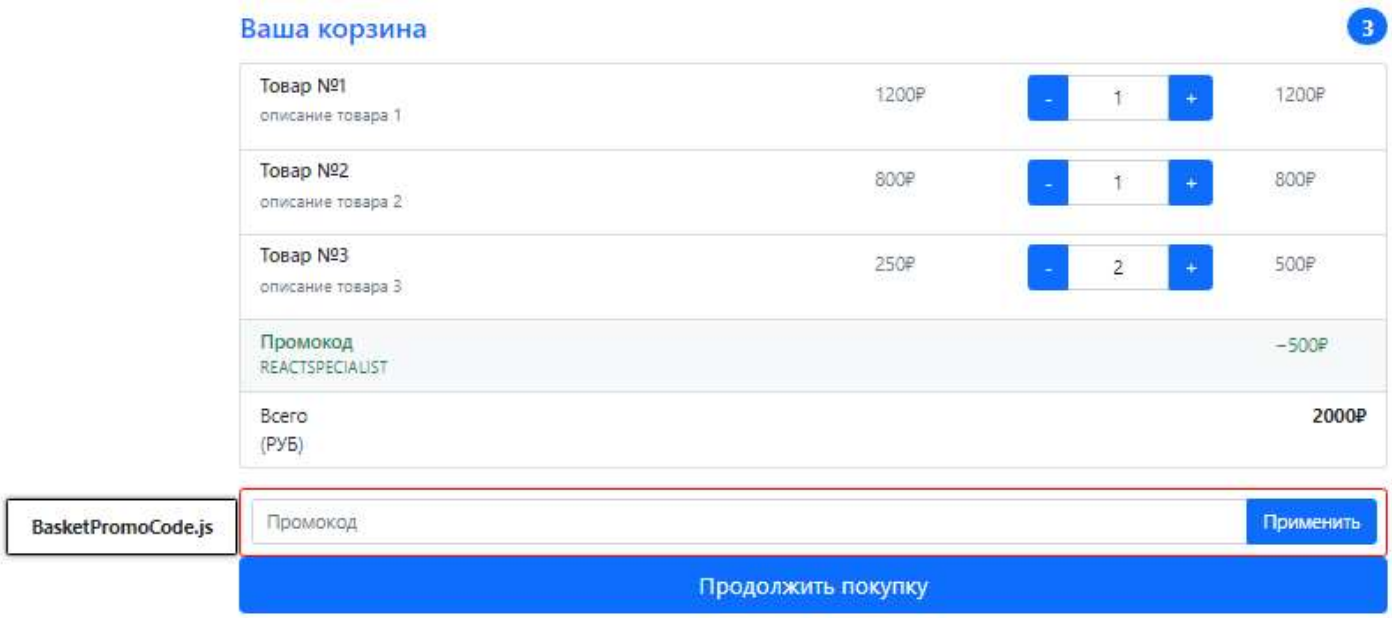
```
.BasketTotal {
  display: flex;
  border: 1px solid rgb(235, 235, 235);
}
.BasketTotal_main {
  width: 93%;
}
.BasketTotal_price .ItemPrice {
  font-weight: bold;
  color: #000;
}
.BasketTotal_title {
  font-size: 1em;
}
```

9.3. Познакомьтесь с утверждениями и убедитесь, что все их них понятны. Если что-то непонятно, спросите преподавателя.

- компонент `BasketTotal` - компонент похожий на `BasketItem` - отображает суммарную стоимость заказа с учетом скидки по промокоду
- `BasketTotal` будет *уметь*
  - выводить суммарную стоиость заказа

Упражнение 10: Создание компонента `BasketPromoCode.js`

В этом упражнении нужно описать содержимое компонента `BasketPromoCode`. На изображении показывается где находится компонент. При создании, подумайте какие свойства/props принимает компонент, как компонент соотносится с другими компонентами



10.1. Измените описание компонента `BasketPromoCode`

```
import React from 'react'
import "./css/BasketPromoCode.css";

import Button from './Button'

const BasketPromoCode = () => {
  return (
    <div className='BasketPromoCode'>
       <input
       className='BasketPromoCode_input'
       placeholder='Промокод'
        />
      <Button value='Применить' onClickHandler={() => alert('Применить')} />
    </div>
  )
}

export default BasketPromoCode
```

10.2. Создайте для него стилевое описание **my_app/components/css/BasketPromoCode.css**

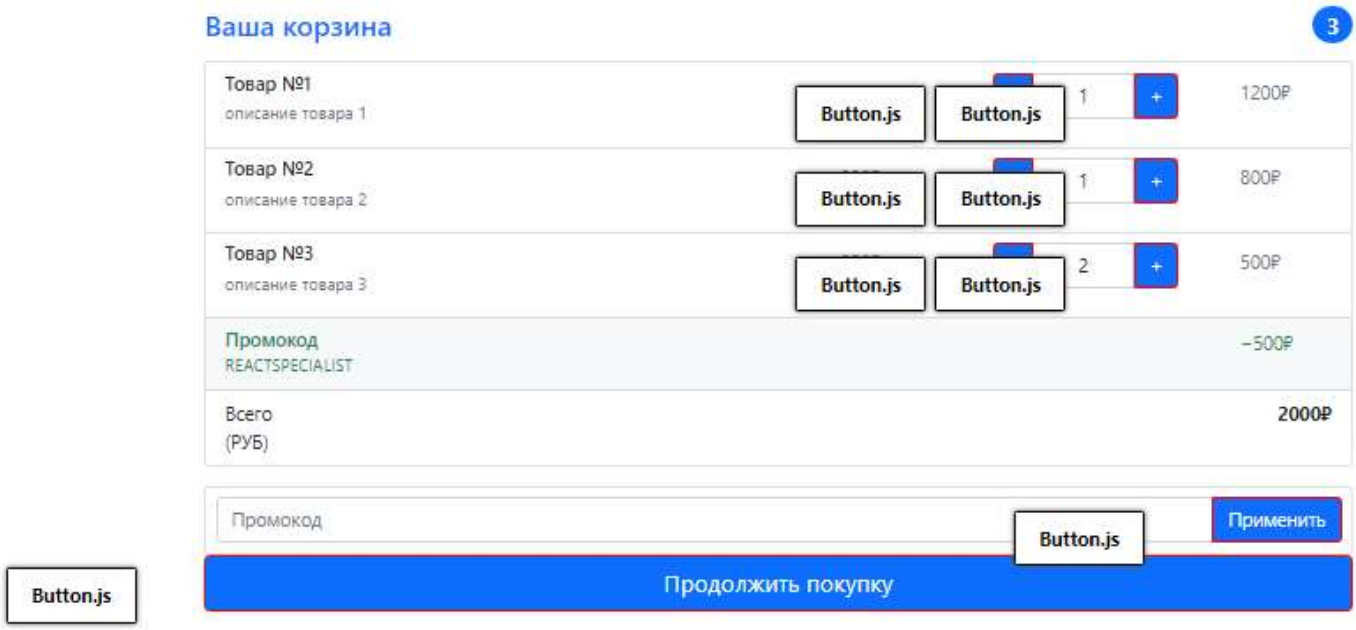
```
.BasketPromoCode {
  display: flex;
  padding: 10px 0;
}
.BasketPromoCode_input {
  width: 90%;
  padding: 10px 15px;
  border-radius: 3px;
  border: 1px solid rgb(235, 235, 235);
  border-radius: 3px 0 0 3px;
}
.BasketPromoCode .btn {
  border-radius: 0 3px 3px 0;
}
.BasketPromoCode_input:focus {
  outline: 1px solid rgb(235, 235, 235);
}
```

10.3. Познакомьтесь с утверждениями и убедитесь, что все их них понятны. Если что-то непонятно, спросите преподавателя.

- компонент `BasketPromoCode` - компонент даёт возможность принять промокод
- `onClickHandler={() => alert('Применить')}` - заглушка на нажатие по кнопке, передается как функция для дальнейшего использования в `Button`

Упражнение 11: Изменение компонента `Button.js`

В этом упражнении нужно описать содержимое компонента `Button`. На изображении показывается где находится компонент. При создании, подумайте какие свойства/props принимает компонент, как компонент соотносится с другими компонентами



11.1. Измените описание компонента `Button`

```
import './css/Button.css';

const Button = ({onClickHandler, value, className}) => {
    return <button className={'btn ' + className} onClick={onClickHandler}>
        {value}
    </button>
}

export default Button
```

11.2. Создайте для него стилевое описание **my_app/components/css/Button.css**

```css
.btn {
  padding: 5px 10px;
  background: rgb(0, 87, 250);
  border: 1px solid rgb(255, 255, 255);
  border-radius: 5px;
  color: rgb(255, 255, 255);
}
.btn:hover {
  background: rgb(58, 127, 255);
  cursor: pointer;
}
.btn-proceed {
  padding: 10px 20px;
  display: block;
  width: 100%;
}
```

11.3. Познакомьтесь с утверждениями и убедитесь, что все их них понятны. Если что-то непонятно, спросите преподавателя.

- компонент `Button` - кнопка
- мы уже работали с `Button`, но тут переписываем в более удобном виде
- `const Button = ({onClickHandler, value, className})` - получаем свойства, в том числе `className`.
  Примечание: Это собственное свойство, потом используем его для изменения `className` в DOM

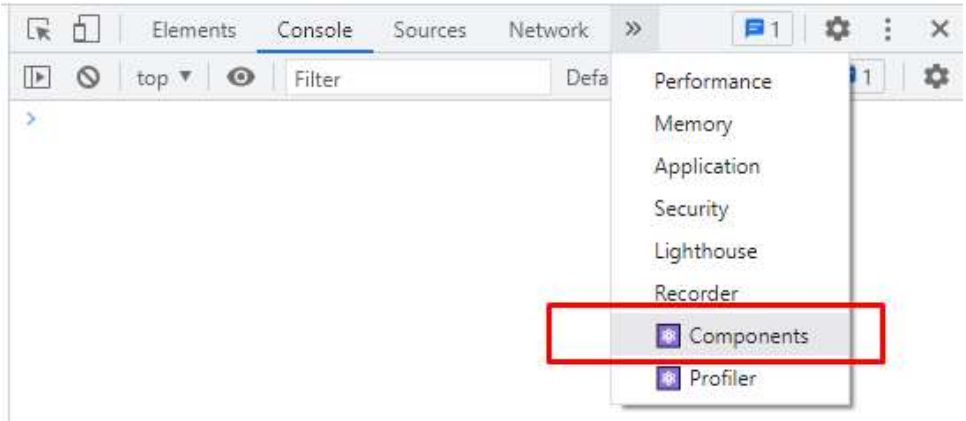
Упражнение 12: Работа с `react devtools` *_опциональная работа_

12.1. Откройте в браузере консоль разработчика через `F12` и зайдите в раздел `Console`

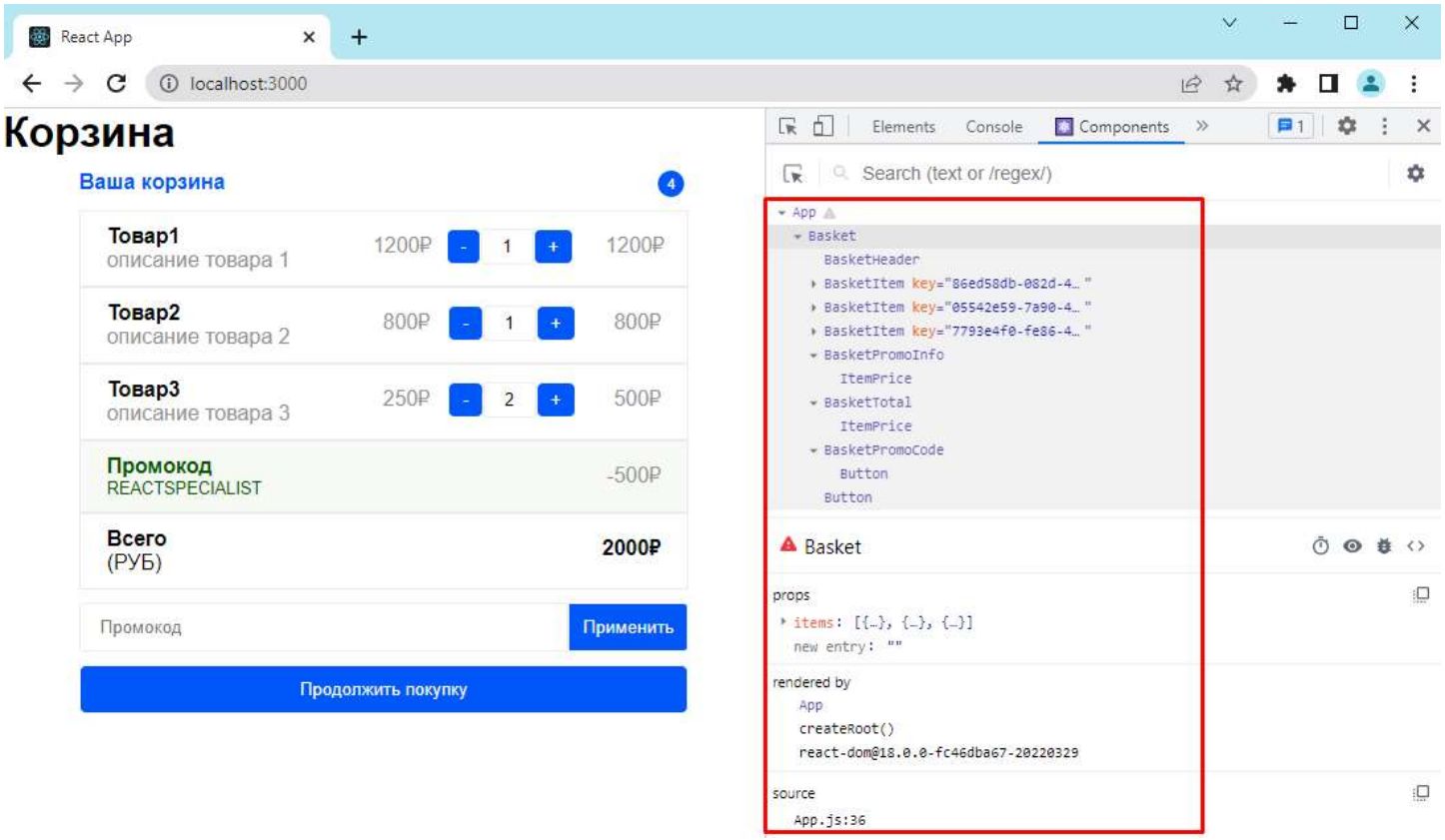
12.2. Найдите ссылку и установите **React Devtools**

12.4. Или установите напрямую react-devtools в Google Chrome (или в Firefox)

12.5. Перезапустите браузер и откройте в консоли пункт **Components**



12.6. **Внимательно** изучите состав компонентов и их свойства/props. Это может очень вам пригодится в дальнейшей работе



12.7. Выводы

- мы создали все компоненты для работы с корзиной
- задали стили этим компонентам (потом найдите время и познакомьтесь с CSS in JS, например styled-components)
- обязательно попробуйте посмотреть под разным углом, импровизируйте
- результатом пратики стал набор компонентов, которые отрисовывают корзину и не имеют (не хранят) состояния