




# Frontender[1.0] JavaScript - Exchange rates APP. Курсы валют. Приложение

|  |   |
|--|---|
| <a href="https://youtu.be/5hcf1QeTAas">🔗</a> YouTube           | <a href="https://youtu.be/5hcf1QeTAas">https://youtu.be/5hcf1QeTAas</a>                       |
| <a href="https://t.me/Dmitry_Kolotilshikov">🔗</a> Telegram     | <a href="https://t.me/Dmitry_Kolotilshikov">https://t.me/Dmitry_Kolotilshikov</a>             |
| <a href="https://github.com/DmitryKolotilshikov/">🔗</a> Github | <a href="https://github.com/DmitryKolotilshikov/">https://github.com/DmitryKolotilshikov/</a> |
| <a href="https://boosty.to/dmitry_ko">🔗</a> Boosty             | <a href="https://boosty.to/dmitry_ko">https://boosty.to/dmitry_ko</a>                         |
| # Номер урока  | 53  |

 **EXCHANGE RATES APP**

 Exchange Rates API:  
  
<https://www.exchangerate-api.com/docs/free>



Сервис [Exchangerate-API](#) предоставляет данные о курсах валют, основываясь на обменных курсах, которые берутся с международных финансовых рынков, а также из данных о валютных торгах и официальных источниках, таких как центральные банки.

### 1. Источник данных

Exchangerate-API обычно агрегирует информацию с различных источников, таких как:

- Центральные банки (например, Европейский центральный банк, Федеральная резервная система США и т.д.).
- Торговые площадки Forex (Foreign Exchange Market).
- Финансовые провайдеры данных, такие как Bloomberg или Reuters.

### 2. Средний курс

Выдаваемый курс часто представляет собой **средний показатель** между курсами покупки и продажи валюты. Это объясняет, почему он может отличаться от курсов, которые вы видите в банке или обменниках, где включаются комиссии и маржа.

### 3. Разные курсы в странах

В каждой стране курсы валют (например, курс покупки и продажи в банке) включают дополнительные факторы, такие как:

- Национальные налоги и сборы.
- Политика конкретного банка или обменника.
- Локальные экономические условия.

Сервис, такой как Exchangerate-API, предоставляет "глобальные" или "справочные" курсы без учета этих региональных наценок.

```
<!DOCTYPE html>
<html lang="en">

<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <link rel="stylesheet" href="styles.css">
  <script src="script.js" type="module" defer></script>
  <title>Exchange rates</title>
</head>

<body>
  <a href="https://www.exchangerate-api.com" style="display: none;">Rates By Exchange Rate
  API</a>

  <div class="container">
    

    <h1 class="title">Exchange rates</h1>

    <section class="info-container">
      <div class="controls-wrapper">
        <select name="firstSelect" class="select-control" data-first-select></select>
        <button data-swap-btn class="swap-btn">swap</button>
        <select name="secondSelect" class="select-control" data-second-select></select>
      </div>
    </section>
  </div>
</body>
</html>
```

```

        </div>

        <div class="controls-wrapper">
            <input type="number" class="input-control" data-first-input>
            <span>=</span>
            <input type="number" class="input-control" data-second-input>
        </div>

        <p data-comparison-info></p>
    </section>
</div>
</body>

</html>

```

```

:root {
    --main-color: #ff7058;
    --bg-color: #b5ded7;
}

* {
    box-sizing: border-box;
}

body {
    margin: 0;
    font-family: 'Segoe UI', Tahoma, Geneva, Verdana, sans-serif;
    background-color: var(--bg-color);
    min-height: 100vh;
    display: flex;
    justify-content: center;
    align-items: flex-start;
}

.container {
    width: 100%;
    max-width: 500px;
    padding: 50px 10px;
    display: flex;
    flex-direction: column;
    justify-content: flex-start;
    align-items: center;
    gap: 16px;
}

.logo {
    width: 160px;
    object-fit: cover;
}

.title {
    margin: 0;
    text-transform: uppercase;
    text-align: center;
    color: var(--main-color);
}

```

```

.info-container {
  display: flex;
  flex-direction: column;
  justify-content: flex-start;
  align-items: center;
  gap: 16px;
}

.controls-wrapper {
  display: flex;
  justify-content: center;
  align-items: center;
  gap: 10px;
}

.select-control {
  font-size: 16px;
  width: 100px;
  padding: 5px;
  background: rgba(255, 255, 255, 0.5);
}

.swap-btn {
  width: 100px;
  height: 30px;
}

.input-control {
  width: 100%;
  font-size: 16px;
  height: 30px;
  background: rgba(255, 255, 255, 0.5);
  border: none;
  outline: none;
}

.input-control:focus-visible {
  box-shadow: 0px 0px 7px 3px rgba(255, 113, 88, 0.8);
}

```

```

// helpers.js
export const roundToFiveDigits = (num) => {
  return parseFloat(num.toFixed(10));
}

```

```

// script.js
import { roundToFiveDigits } from "./helpers.js";

const firstSelect = document.querySelector("[data-first-select]");
const secondSelect = document.querySelector("[data-second-select]");

const swapBtn = document.querySelector("[data-swap-btn]");
const comparisonInfo = document.querySelector("[data-comparison-info]");

const firstInput = document.querySelector("[data-first-input]");
const secondInput = document.querySelector("[data-second-input]");

```

```

const BASE_URL = "https://open.er-api.com/v6/latest";
const FIRST_DEFAULT_CURRENCY = "USD";
const SECOND_DEFAULT_CURRENCY = "BYN";

let rates = {};

// select events
firstSelect.addEventListener("change", () => updateExchangeRates());
secondSelect.addEventListener("change", () => renderInfo());

// input events
firstInput.addEventListener("input", () => {
  secondInput.value = roundToFiveDigits(firstInput.value * rates[secondSelect.value]);
})
secondInput.addEventListener("input", () => {
  firstInput.value = roundToFiveDigits(secondInput.value / rates[secondSelect.value]);
})

// swap button event
swapBtn.addEventListener("click", () => {
  const temp = firstSelect.value;
  firstSelect.value = secondSelect.value;
  secondSelect.value = temp;

  updateExchangeRates();
})

// update exchange rates
const updateExchangeRates = async () => {
  try {
    const response = await fetch(`${BASE_URL}/${firstSelect.value}`);
    const data = await response.json();

    rates = data.rates;
    renderInfo();
  } catch (error) {
    console.error(error.message);
  }
}

// render info
const renderInfo = () => {
  comparisonInfo.textContent = `1 ${firstSelect.value} = ${rates[secondSelect.value]} ${secondSelect.value}`;

  firstInput.value = rates[firstSelect.value];
  secondInput.value = rates[secondSelect.value];
}

// populate selects
const populateSelect = () => {
  firstSelect.innerHTML = "";
  secondSelect.innerHTML = "";

  for (const currency of Object.keys(rates)) {
    firstSelect.innerHTML += `
      <option value="${currency}" ${currency === FIRST_DEFAULT_CURRENCY ? "selected" :

```

```

    ">${currency}</option>
    ,
    secondSelect.innerHTML += `
        <option value="${currency}" ${currency === SECOND_DEFAULT_CURRENCY ? "selected" :
    ">${currency}</option>
    ,
    }
}

// getInitialRates - initial fn to populate selects
const getInitialRates = async () => {
    try {
        const response = await fetch(`${BASE_URL}/${FIRST_DEFAULT_CURRENCY}`);
        const data = await response.json();

        rates = data.rates;
        populateSelect();
        renderInfo();
    } catch (error) {
        console.error(error.message);
    }
}

getInitialRates();

```