

---

ДИСЦИПЛИНА	Фронтенд и бэкенд разработка
ИНСТИТУТ	ИПТИП
КАФЕДРА	Индустриального программирования
ВИД УЧЕБНОГО МАТЕРИАЛА	Методические указания к практическим занятиям
ПРЕПОДАВАТЕЛЬ	Астафьев Рустам Уралович
СЕМЕСТР	1 семестр, 2025/2026 уч. год

---

Ссылка на материал:

<https://github.com/astafiev-rustam/frontend-and-backend-development/tree/practice-1-23>

---

## Практическое занятие 23: React роутинг: использование маршрутизации и параметров

---

В рамках данного занятия будут рассмотрены возможности маршрутизации и параметров маршрутизации. Подробную информацию об этом можно найти в материалах лекций, а также в материалах:

<https://ru.hexlet.io/blog/posts/react-router-v6>

<https://metanit.com/web/react/4.1.php>

### Теоретическая часть

Пример 1. Базовая настройка React Router

**Проблема:** Нужно создать многостраничное приложение с навигацией между разными разделами без перезагрузки страницы.

**Подход к решению:** Используем React Router для настройки маршрутов и компоненты Link для навигации.

**Исходный код в файле App.js:**

```
import { BrowserRouter as Router, Routes, Route, Link } from 'react-router-dom';
import Home from './pages/Home';
import About from './pages/About';
import Contact from './pages/Contact';
import './App.css';

function App() {
  return (
    <Router>
      <div className="app">
        {/* Навигационное меню */}
      </div>
    </Router>
  );
}

export default App;
```

```
<nav className="main-nav">
  <div className="nav-brand">
    <h2>Moe Приложение</h2>
  </div>
  <ul className="nav-links">
    <li>
      <Link to="/">Главная</Link>
    </li>
    <li>
      <Link to="/about">О нас</Link>
    </li>
    <li>
      <Link to="/contact">Контакты</Link>
    </li>
  </ul>
</nav>

{/* Основное содержимое */}
<main className="main-content">
  <Routes>
    <Route path="/" element={<Home />} />
    <Route path="/about" element={<About />} />
    <Route path="/contact" element={<Contact />} />
  </Routes>
</main>
</div>
</Router>
);
}

export default App;
```

## Страница главная в файле pages/Home.jsx:

```
function Home() {
  return (
    <div className="page">
      <h1>Добро пожаловать на главную страницу!</h1>
      <p>Это стартовая страница нашего приложения.</p>
      <div className="features">
        <h2>Наши возможности:</h2>
        <ul>
          <li>Навигация между страницами</li>
          <li>Динамическая загрузка контента</li>
          <li>Быстрая работа без перезагрузки</li>
        </ul>
      </div>
    </div>
  );
}

export default Home;
```

## Страница "О нас" в файле pages/About.jsx:

```
function About() {
  return (
    <div className="page">
      <h1>О нашем приложении</h1>
      <p>Это учебное приложение создано для изучения React Router.</p>
      <div className="about-content">
        <h2>Наша миссия</h2>
        <p>Помогать разработчикам изучать современные технологии веб-разработки.</p>
      </div>

      <h2>Технологии</h2>
      <ul>
        <li>React</li>
        <li>React Router</li>
        <li>JavaScript ES6+</li>
      </ul>
    </div>
  );
}

export default About;
```

## Страница "Контакты" в файле pages/Contact.jsx:

```
function Contact() {
  return (
    <div className="page">
      <h1>Наши контакты</h1>
    </div>
  );
}

export default Contact;
```

## Пример 2. Динамические маршруты с параметрами

**Проблема:** Нужно создавать страницы для разных пользователей, используя один компонент, но с разными данными.

**Подход к решению:** Используем параметры в маршрутах и хук useParams для их получения.

**Обновленный App.js с динамическими маршрутами:**

```
import React from 'react';
import { BrowserRouter as Router, Routes, Route, Link } from 'react-router-dom';
import Home from './pages/Home';
import About from './pages/About';
import UserProfile from './pages/UserProfile';
import './App.css';

function App() {
    // Пример данных пользователей
    const users = [
        { id: 1, name: 'Анна' },
        { id: 2, name: 'Иван' },
        { id: 3, name: 'Мария' }
    ];

    return (
        <Router>
            <div className="app">
                <nav className="main-nav">
                    <h2>Трекер технологий</h2>
                    <ul className="nav-links">
                        <li><Link to="/">Главная</Link></li>
                        <li><Link to="/about">0 проекте</Link></li>
                        <li>
                            <span>Пользователи:</span>
                            <ul>
                                {users.map(user => (
                                    <li><Link key={user.id} to={`/user/${user.id}`}>{user.name}</Link>
                                    </li>
                                )));
                            </ul>
                        </li>
                    </ul>
                </nav>

                <main className="main-content">
                    <Routes>
                        <Route path="/" element={<Home />} />
                        <Route path="/about" element={<About />} />
                        {/* Динамический маршрут для пользователей */}
                        <Route path="/user/:userId" element={<UserProfile />} />
                    </Routes>
                </main>
            </div>
        </Router>
    );
}

export default App;
```

**Компонент профиля пользователя в pages/UserProfile.jsx:**

```
import { useParams, Link } from 'react-router-dom';

function UserProfile() {
    // Получаем параметр userId из URL
    const { userId } = useParams();

    // В реальном приложении здесь был бы запрос к API
    // Сейчас используем mock данные
    const users = {
        1: { id: 1, name: 'Анна', role: 'Фронтенд разработчик', progress: 75 },
        2: { id: 2, name: 'Иван', role: 'Бэкенд разработчик', progress: 60 },
        3: { id: 3, name: 'Мария', role: 'Fullstack разработчик', progress: 85 }
    };

    const user = users[userId];

    // Если пользователь не найден
    if (!user) {
        return (
            <div className="page">
                <h1>Пользователь не найден</h1>
                <p>Пользователь с ID {userId} не существует.</p>
                <Link to="/">Вернуться на главную</Link>
            </div>
        );
    }

    return (
        <div className="page">
            <h1>Профиль пользователя</h1>
            <div className="user-info">
                <h2>{user.name}</h2>
                <p><strong>Должность:</strong> {user.role}</p>
                <p><strong>Прогресс:</strong> {user.progress}%</p>
            </div>

            <div className="user-actions">
                <Link to="/" className="back-link">← Назад к списку</Link>
            </div>
        </div>
    );
}

export default UserProfile;
```

**Пример 3. Программная навигация и защищенные маршруты**

**Проблема:** Нужно ограничить доступ к некоторым страницам только для авторизованных пользователей и реализовать перенаправления.

**Подход к решению:** Создаем компонент-обертку для защищенных маршрутов и используем хук useNavigate для программной навигации.

### Компонент логина в pages/Login.jsx:

```
import { useState } from 'react';
import { useNavigate } from 'react-router-dom';

function Login({ onLogin }) {
  const [username, setUsername] = useState('');
  const [password, setPassword] = useState('');
  const navigate = useNavigate();

  const handleSubmit = (e) => {
    e.preventDefault();

    if (username === 'admin' && password === 'password') {
      localStorage.setItem('isLoggedIn', 'true');
      localStorage.setItem('username', username);

      // Вызываем колбэк для обновления состояния в App
      onLogin(username);

      // Перенаправляем на главную
      navigate('/');
    } else {
      alert('Неверные данные для входа');
    }
  };

  return (
    <div className="page">
      <h1>Вход в систему</h1>
      <form onSubmit={handleSubmit} className="login-form">
        <div className="form-group">
          <label>Имя пользователя:</label>
          <input
            type="text"
            value={username}
            onChange={(e) => setUsername(e.target.value)}
            required
          />
        </div>

        <div className="form-group">
          <label>Пароль:</label>
          <input
            type="password"
            value={password}
            onChange={(e) => setPassword(e.target.value)}
            required
          />
        </div>
      </form>
    </div>
  );
}

export default Login;
```

```

        </div>

        <button type="submit">Войти</button>
    </form>
</div>
);
}

export default Login;

```

### Компонент-обертка для защищенных маршрутов в `components/ProtectedRoute.jsx`:

```

import { Navigate } from 'react-router-dom';

function ProtectedRoute({ children, isLoggedIn }) {
    // Используем переданное состояние вместо прямого чтения localStorage
    if (!isLoggedIn) {
        return <Navigate to="/login" replace />;
    }

    return children;
}

export default ProtectedRoute;

```

### Обновленный `App.js` с защищенными маршрутами:

```

import { BrowserRouter as Router, Routes, Route, Link } from 'react-router-dom';
import { useState, useEffect } from 'react'; // Добавляем useEffect
import Home from './pages/Home';
import About from './pages/About';
import Login from './pages/Login';
import Dashboard from './pages/Dashboard';
import ProtectedRoute from './components/ProtectedRoute';
import './App.css';

function App() {
    // Состояние для отслеживания авторизации
    const [isLoggedIn, setIsLoggedIn] = useState(false);
    const [username, setUsername] = useState('');

    // Проверяем авторизацию при загрузке и при изменении
    useEffect(() => {
        const loggedIn = localStorage.getItem('isLoggedIn') === 'true';
        const user = localStorage.getItem('username') || '';
        setIsLoggedIn(loggedIn);
        setUsername(user);
    }, []);
}

```

```
const handleLogin = (user) => {
  setIsLoggedIn(true);
  setUsername(user);
};

const handleLogout = () => {
  localStorage.removeItem('isLoggedIn');
  localStorage.removeItem('username');
  setIsLoggedIn(false);
  setUsername('');
};

return (
  <Router>
    <div className="app">
      <nav className="main-nav">
        <h2>Трекер технологий</h2>
        <ul className="nav-links">
          <li><Link to="/">Главная</Link></li>
          <li><Link to="/about">О проекте</Link></li>

          {isLoggedIn ? (
            <>
              <li><Link to="/dashboard">Панель управления</Link></li>
              <li className="user-info">
                <span>Привет, {username}!</span>
                <button onClick={handleLogout} className="logout-btn">
                  Выйти
                </button>
              </li>
            </>
          ) : (
            <li><Link to="/login">Войти</Link></li>
          )}
        </ul>
      </nav>

      <main className="main-content">
        <Routes>
          <Route path="/" element={<Home />} />
          <Route path="/about" element={<About />} />
          <Route
            path="/login"
            element={<Login onLogin={handleLogin} />}
          />

          <Route
            path="/dashboard"
            element={
              <ProtectedRoute isLoggedIn={isLoggedIn}>
                <Dashboard />
              </ProtectedRoute>
            }
          />
        </Routes>
      </main>
    </div>
  </Router>
)
```

```
        </Routes>
      </main>
    </div>
  </Router>
);
}

export default App;
```

## Практическая часть

Добавление маршрутизации в трекер технологий

### Шаг 1: Установите React Router

```
npm install react-router-dom
```

### Шаг 2: Создайте структуру папок для страниц

```
src/
├── pages/
│   ├── Home.js
│   ├── TechnologyList.js
│   ├── TechnologyDetail.js
│   └── AddTechnology.js
└── components/
    ├── Navigation.js
    └── TechnologyCard.js
└── App.js
```

### Шаг 3: Создайте компонент навигации

```
// components/Navigation.js
import { Link, useLocation } from 'react-router-dom';

function Navigation() {
  const location = useLocation();

  return (
    <nav className="main-navigation">
      <div className="nav-brand">
        <Link to="/">
          <h2>🚀 Трекер технологий</h2>
        </Link>
      </div>

      <ul className="nav-menu">
```

```
<li>
  <Link
    to="/"
    className={location.pathname === '/' ? 'active' : ''}
  >
  Главная
</Link>
</li>
<li>
  <Link
    to="/technologies"
    className={location.pathname === '/technologies' ? 'active' : ''}
  >
  Все технологии
</Link>
</li>
<li>
  <Link
    to="/add-technology"
    className={location.pathname === '/add-technology' ? 'active' : ''}
  >
  Добавить технологию
</Link>
</li>
</ul>
</nav>
);
}

export default Navigation;
```

#### Шаг 4: Создайте страницу со списком технологий

```
// pages/TechnologyList.js
import { Link } from 'react-router-dom';
import { useState, useEffect } from 'react';

function TechnologyList() {
  const [technologies, setTechnologies] = useState([]);

  // Загружаем технологии из localStorage
  useEffect(() => {
    const saved = localStorage.getItem('technologies');
    if (saved) {
      setTechnologies(JSON.parse(saved));
    }
  }, []);

  return (
    <div className="page">
      <div className="page-header">
        <h1>Все технологии</h1>
      </div>
      <ul>
        {technologies.map(tech => (
          <li>
            <Link
              to={`/technologies/${tech.id}`}
              className={location.pathname === `/technologies/${tech.id}` ? 'active' : ''}
            >
              {tech.name}
            </Link>
          </li>
        ))}
      </ul>
    </div>
  );
}

export default TechnologyList;
```

```

<Link to="/add-technology" className="btn btn-primary">
    + Добавить технологию
</Link>
</div>

<div className="technologies-grid">
{technologies.map(tech => (
    <div key={tech.id} className="technology-item">
        <h3>{tech.title}</h3>
        <p>{tech.description}</p>
        <div className="technology-meta">
            <span className={`status status-${tech.status}`}>
                {tech.status}
            </span>
            <Link to={`/technology/${tech.id}`} className="btn-link">
                Подробнее →
            </Link>
        </div>
    </div>
))
</div>

{technologies.length === 0 && (
    <div className="empty-state">
        <p>Технологий пока нет.</p>
        <Link to="/add-technology" className="btn btn-primary">
            Добавить первую технологию
        </Link>
    </div>
)
);
}

export default TechnologyList;

```

## Шаг 5: Создайте страницу деталей технологии

```

// pages/TechnologyDetail.js
import { useParams, Link, useNavigate } from 'react-router-dom';
import { useState, useEffect } from 'react';

function TechnologyDetail() {
    const { techId } = useParams();
    const navigate = useNavigate();
    const [technology, setTechnology] = useState(null);

    useEffect(() => {
        const saved = localStorage.getItem('technologies');
        if (saved) {
            const technologies = JSON.parse(saved);
            const tech = technologies.find(t => t.id === parseInt(techId));
            setTechnology(tech);
        }
    }, []);
}

export default TechnologyDetail;

```

```
        setTechnology(tech);
    }
}, [techId]);

const updateStatus = (newStatus) => {
    const saved = localStorage.getItem('technologies');
    if (saved) {
        const technologies = JSON.parse(saved);
        const updated = technologies.map(tech =>
            tech.id === parseInt(techId) ? { ...tech, status: newStatus } : tech
        );
        localStorage.setItem('technologies', JSON.stringify(updated));
        setTechnology({ ...technology, status: newStatus });
    }
};

if (!technology) {
    return (
        <div className="page">
            <h1>Технология не найдена</h1>
            <p>Технология с ID {techId} не существует.</p>
            <Link to="/technologies" className="btn">
                ← Назад к списку
            </Link>
        </div>
    );
}

return (
    <div className="page">
        <div className="page-header">
            <Link to="/technologies" className="back-link">
                ← Назад к списку
            </Link>
            <h1>{technology.title}</h1>
        </div>

        <div className="technology-detail">
            <div className="detail-section">
                <h3>Описание</h3>
                <p>{technology.description}</p>
            </div>

            <div className="detail-section">
                <h3>Статус изучения</h3>
                <div className="status-buttons">
                    <button
                        onClick={() => updateStatus('not-started')}
                        className={technology.status === 'not-started' ? 'active' : ''}
                    >
                        Не начато
                    </button>
                    <button
                        onClick={() => updateStatus('in-progress')}
                    >
                        В процессе
                    </button>
                </div>
            </div>
        </div>
    </div>
);
```

```
        className={technology.status === 'in-progress' ? 'active' : ''}
      >
        В процессе
      </button>
      <button
        onClick={() => updateStatus('completed')}
        className={technology.status === 'completed' ? 'active' : ''}
      >
        Завершено
      </button>
    </div>
  </div>

  {technology.notes && (
    <div className="detail-section">
      <h3>Мои заметки</h3>
      <p>{technology.notes}</p>
    </div>
  )}
  </div>
</div>
);

}

export default TechnologyDetail;
```

## Самостоятельная работа

**Задание 1:** Создайте страницу "Статистика" с графиком прогресса

**Задание 2:** Добавьте страницу "Настройки" для управления приложением

### Что проверить перед завершением:

- Навигация между страницами работает без перезагрузки
- Параметры в URL правильно обрабатываются
- Защищенные маршруты перенаправляют неавторизованных пользователей
- Данные сохраняются между переходами по страницам

Теперь ваше приложение стало полноценным SPA с навигацией и разными страницами!