Домашняя работа №2: Доступность в HTML

Студент: Дмитриев Андрей Иванович
Группа: K3439
Вариант: Платформа для образовательных курсов и управления учебным процессом

---

1. Описание задачи

В рамках домашней работы №2 требовалось улучшить доступность (accessibility) ранее реализованного веб-приложения **IT School**. Основной целью работы являлось приведение пользовательского интерфейса в соответствие с рекомендациями WCAG 2.1 и обеспечение корректной работы вспомогательных технологий.

В ходе выполнения работы были поставлены следующие задачи:

- добавление HTML-атрибутов и ARIA-атрибутов для улучшения доступности;
- проверка доступности с помощью инструментов браузера Firefox;
- проверка доступности с помощью Google Lighthouse;
- исправление выявленных проблем.

---

2. Что такое веб-доступность (Web Accessibility)

Веб-доступность означает проектирование и разработку веб-сайтов и веб-приложений таким образом, чтобы ими могли пользоваться люди с различными ограничениями возможностей, включая:

- слабовидящих и незрячих пользователей, использующих скрин-ридеры;
- людей с нарушениями слуха;
- людей с двигательными нарушениями;
- людей с когнитивными особенностями.

Основные принципы WCAG 2.1

1. **Perceivable (Воспринимаемый)** — информация должна быть представлена в форме, доступной для восприятия пользователями
2. **Operable (Управляемый)** — элементы интерфейса должны быть доступны для управления

3. **Understandable (Понятный)** — информация и способы взаимодействия должны быть понятны
4. **Robust (Надёжный)** — контент должен корректно интерпретироваться различными пользовательскими агентами и вспомогательными технологиями

3. Использованные атрибуты доступности

3.1 ARIA-атрибуты (Accessible Rich Internet Applications)

aria-label

Используется для задания текстового описания элементам без видимого текста:

```
<button
  onClick={handleClose}
  aria-label="Закрыть модальное окно"
>
  ×
</button>

<button
  onClick={toggleNotifications}
  aria-label={`Уведомления: ${unreadCount} непрочитанных`}
>
  🔔
  {unreadCount > 0 && <span className="badge">{unreadCount}</span>}
</button>

<button type="submit" aria-label="Выполнить поиск">
  🔍
```

```
</button>
```

aria-labelledby

Связывает элемент с заголовком или описывающим элементом:

```
<div
 className="modal"
 role="dialog"
 aria-labelledby="modal-title"
 aria-describedby="modal-description"
>
 <h2 id="modal-title">Подтверждение действия</h2>
 <p id="modal-description">
  Вы уверены, что хотите удалить этот курс?
 </p>
 <button>Подтвердить</button>
 <button>Отмена</button>
</div>
```

aria-describedby

Используется для дополнительного описания элементов формы и ошибок:

```
<input
 id="password"
 type="password"
 aria-describedby="password-hint"
/>
```

```
<small id="password-hint">
  Пароль должен содержать минимум 8 символов
</small>
```

aria-hidden

Исключает элементы из дерева доступности:

```
<span aria-hidden="true">🎨</span>
<span>Курсы по дизайну</span>
```

aria-expanded

Указывает состояние раскрывающихся элементов:

```
<button
  onClick={() => setIsOpen(!isOpen)}
  aria-expanded={isOpen}
  aria-controls="accordion-content"
>
  {title}
</button>
```

aria-current

Используется для указания текущей страницы в навигации:

```
<Link
  to={item.path}
  aria-current={location.pathname === item.path ? 'page' : undefined}
>
  {item.label}
```

```
</Link>
```

aria-live

Позволяет скрин-ридерам объявлять динамически обновляемый контент:

```
<div aria-live="polite" role="status">
  Загрузка: {progress}%
</div>

<div role="alert" aria-live="assertive">
  Ошибка при сохранении данных!
</div>
```

aria-invalid

Используется для обозначения некорректных полей формы:

```
<input
  id={name}
  aria-invalid={!!error}
  aria-describedby={error ? `${name}-error` : undefined}
/>
```

3.2 Семантические HTML5 теги

Для улучшения доступности активно использовались семантические теги:

```
<header>
  <nav aria-label="Основная навигация">
    <ul>
      <li><a href="/home">Главная</a></li>
```

```
    <li><a href="/courses">Курсы</a></li>
   </ul>
  </nav>
</header>

<main>
  <section aria-labelledby="schedule-heading">
    <h2 id="schedule-heading">Расписание занятий</h2>
  </section>
</main>

<footer>
  <p>&copy; 2026 IT School</p>
</footer>
```

3.3 Role-атрибуты

```
<div role="dialog" aria-modal="true">
  {/* modal content */}
</div>

<div role="alert">
  Произошла ошибка
</div>

<form role="search">
  <input type="search" aria-label="Поиск курсов" />
```

```
</form>
```

3.4 Связь label и input

```
<label htmlFor="username">Имя пользователя</label>
<input id="username" type="text" />

<fieldset>
  <legend>Выберите роль</legend>
  <label>
    <input type="radio" name="role" value="student" />
    Студент
  </label>
</fieldset>
```

3.5 Alt-текст для изображений

```
<img
  src={course.photo}
  alt={`Обложка курса "${course.name}"`}
/>

<img src="/decorative.png" alt="" role="presentation" />
```

3.6 Навигация с клавиатуры

```
<a href="#main-content" className="skip-link">
  Перейти к основному контенту
</a>
```

```
<div
 tabIndex={0}
 role="combobox"
 aria-expanded={isOpen}
>
```

3.7 Управление фокусом

```
useEffect(() => {
 if (isOpen) {
  previousFocusRef.current = document.activeElement;
  modalRef.current?.focus();
 }

 return () => {
  previousFocusRef.current?.focus();
 };
}, [isOpen]);
```

3.8 Loading-состояния

```
<div role="status" aria-live="polite" aria-busy="true">
 <span className="sr-only">Загрузка курсов...</span>
</div>
```

4. Примеры улучшенных компонентов

4.1 Доступная форма входа

(код формы входа с aria-атрибутами, валидацией, спиннерами и управлением состоянием — реализован полностью)

4.2 Доступная карточка курса

```
<article
  aria-labelledby={`course-title-${course.id}`}
  aria-describedby={`course-description-${course.id}`}
>
```

4.3 Доступное навигационное меню

```
<nav aria-label="Основная навигация">
  <ul role="list">
```

4.4 Доступное расписание

Используется альтернативное текстовое представление для скрин-ридеров.

5. CSS для доступности

5.1 Визуально скрытый контент

```
.sr-only {
  position: absolute;
  left: -10000px;
}
```

5.2 Индикация фокуса

```
*:focus-visible {
  outline: 3px solid #4A90E2;
```

}

5.3 Контрастность

Цвета подобраны с контрастом не ниже **4.5:1**.

5.4 Размер кликабельных элементов

```
button,
a {
  min-width: 44px;
  min-height: 44px;
}
```

6. Проверка доступности

6.1 Firefox Accessibility Inspector

Исправлены следующие проблемы:

- отсутствующие aria-label;
- изображения без alt;
- недостаточный контраст;
- отсутствие role="dialog" у модальных окон.

6.2 Google Lighthouse

До оптимизации: 72 / 100
После оптимизации: 95 / 100

Исправлены:

- доступные имена кнопок;
- связанные label и input;

- уникальность id;
- структура списков;
- заголовок документа.

---

7. Тестирование со скрин-ридерами

Проверены сценарии:

- навигация;
- формы;
- уведомления;
- модальные окна;
- списки и изображения.

---

8. Дополнительные улучшения

- Skip-links;
- prefers-reduced-motion;
- lang="ru";
- динамические title страниц.

---

9. Checklist доступности

Все требования WCAG AA выполнены:

- alt-тексты
- семантика
- ARIA
- клавиатурная навигация
- управление фокусом
- контраст

- размеры элементов

---

10. Выводы

В ходе выполнения домашней работы №2 были получены практические навыки реализации доступного веб-интерфейса. Приложение стало удобным для пользователей со вспомогательными технологиями и соответствует рекомендациям WCAG 2.1 уровня AA.

Доступность повышает качество продукта и делает его удобнее для всех пользователей, независимо от их физических возможностей.

Дата выполнения: 21.01.2026