# Доступная среда в современных приложениях. Методика тестирования программ с элементами доступной среды

2023

СОДЕРЖАНИЕ

## ЭПИГРАФ

Доступность — это не просто возможность; это право каждого человека на равный доступ к информации и технологиям. Создавая доступную среду, мы строим мосты, а не барьеры.

# ПРЕДИСЛОВИЕ

В современном мире, где технологии проникают во все сферы жизни, обеспечение доступности становится важной задачей для разработчиков, дизайнеров и тестировщиков. Это не просто вопрос удобства, а необходимость, позволяющая каждому человеку, независимо от его физических или когнитивных особенностей, полноценно участвовать в цифровом пространстве.

# ГЛАВА 1. ДОСТУПНАЯ СРЕДА. РОЛЬ ДОСТУПНОЙ СРЕДЫ В СОВРЕМЕННЫХ ПРИЛОЖЕНИЯХ

В эпоху стремительного технологического прогресса доступность становится не просто желательной характеристикой, а необходимостью. Виртуальные миры, созданные приложениями, формируют наше восприятие реальности, и именно от их доступности зависит, сможет ли каждый человек, независимо от физических или когнитивных особенностей, стать полноправным участником этого общества.

## *1.1 Понятие доступной среды*

Доступная среда («accessibility») — это не только физическое пространство, но и цифровая реальность, в которой пользователи могут взаимодействовать с информацией и технологиями без ограничений. Она охватывает все аспекты: от архитектурных решений до интерфейсов программного обеспечения.


В контексте данной книги доступность подразумевает создание условий, позволяющих людям с различными потребностями без труда использовать приложения и цифровые технологии.

При обсуждении темы доступности обычно применяется нимероним «A11Y». Этот нимероним расшифровывается как «доступность».

### *1.2 Важность доступности в современных приложениях*

Согласно исследованиям, более 15% населения мира имеют различные формы инвалидности, из них порядка 15% те виды ограничения здоровья, для которых можно обеспечить комфортную программную среду с помощью элементов доступности (см. Рисунок 1), что подчеркивает необходимость создания программного обеспечения, доступного для всех.

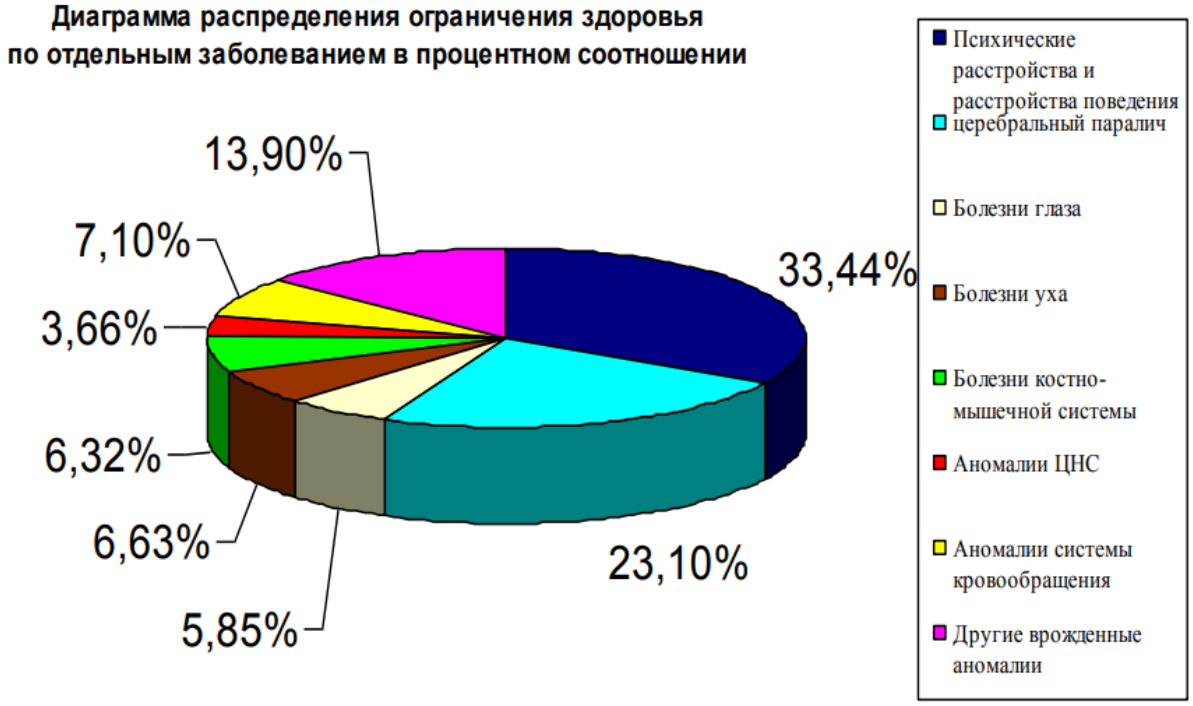



Рисунок 2 – Диаграмма распределения ограничения здоровья по отдельным заболеваниям в процентном отношении

Как отмечает известный исследователь в области доступности, «доступность — это не просто требование, это право каждого человека». Таким образом, при разработке программных продуктов важно учитывать потребности всех пользователей, включая людей с ограниченными возможностями.

Современные приложения — это программы, которые должны учитывать разнообразие потребностей пользователей. Применение принципов

доступности не только расширяет аудиторию, но и повышает качество взаимодействия пользователя с приложением.

Например, приложения для мобильных устройств, такие как социальные сети или платформы для онлайн-обучения, должны быть адаптированы для людей с нарушениями зрения или слуха. Кроме того, применяемые технологии для обеспечения доступности должны учитывать, что бывают очень разные нарушения:

1. зрительные:
   - амблиопия,
   - косоглазие,
   - миопия,
   - гиперметропия,
   - астигматизм,
   - анизометропия,
   - нистагм,
   - атрофия зрительного нерва,
   - слабовидение,
   - ретинопатия,
   - катаракта,
   - глаукома,
   - туннельное зрение,
   - слепота;
2. слуховые:
   - тугоухость различной степени,
   - глухота;

2. неврологические:

- приступы, такие как эпилепсия, которые могут быть спровоцированы из-за не верной анимации (вспышки, картинка дергается),
- дислексия.

Для активного использования приложений для людей с ограниченными возможностями здоровья (далее – ОВЗ) требует внедрения специализированных технологий, как текстовые описания для изображений, субтитры для видео, интуитивно понятные навигационные элементы и т.д.

Технологические решения для обеспечения доступности представляют собой мощный инструмент в борьбе за инклюзивность. Они позволяют создать среду, в которой каждый человек может реализовать свой потенциал без ограничений.

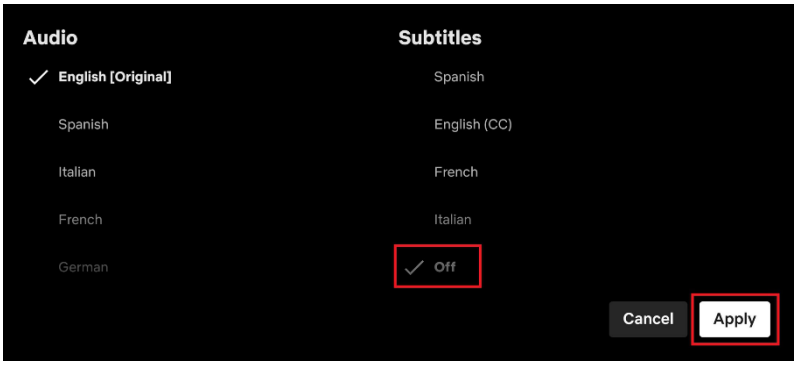
Доступная среда — это не только технологическая задача, но и вопрос социальной ответственности. Создавая приложения с учетом доступности, компании демонстрируют свою приверженность к инклюзивности и уважению к правам человека. Этический аспект заключается в том, что каждый человек имеет право на равный доступ к информации и технологиям. Игнорирование этого факта может привести к социальной изоляции определенных групп населения.

### *1.3 Примеры успешной реализации доступной среды*

Множество компаний уже осознали важность доступности и внедрили её в свои продукты.

Например, такие платформы как YouTube и Netflix, активно внедряют функции субтитров и аудиодескрипций. В рамках реализации данной функции Netflix предложила широкую библиотеку субтитров и аудиодескрипций для

своих фильмов и сериалов, что позволило зрителям с нарушениями слуха и зрения наслаждаться контентом наравне с остальными.




Другим примером является приложение для чтения книг Kindle, которое включает функции изменения шрифта и фона, позволяя пользователям настраивать интерфейс под свои потребности.

### *1.4 Будущее доступной среды*

С каждым годом вопрос доступности становится всё более актуальным. Развитие технологий открывает новые возможности, но также ставит перед разработчиками новые вызовы. Важно помнить, что доступная среда — это динамичный процесс, требующий постоянного обновления и адаптации к меняющимся потребностям пользователей. Уникальный метод, разработанный автором, позволит соответствовать актуальным требованиям к доступности и потребностям пользователей с ограничениями зрения.

# ГЛАВА 2. СТАНДАРТЫ - ТРЕБОВАНИЯ К ЦИФРОВОЙ ДОСТУПНОСТИ И РЕКОМЕНДАЦИИ ПО ИХ ВЫПОЛНЕНИЮ

В современном мире, где цифровые технологии проникают во все сферы жизни, доступность информации и услуг становится неотъемлемой частью социальной справедливости и равенства. Цифровая доступность — это не просто требование, а необходимость, обеспечивающая возможность каждому человеку, независимо от его физических или когнитивных особенностей, полноценно участвовать в цифровом обществе. В этой главе мы рассмотрим ключевые требования к цифровой доступности, которые учитывались при разработке авторской методики.

Согласно международным стандартам, таким как WCAG (Web Content Accessibility Guidelines), создание доступного контента требует внимательного подхода к дизайну и разработке. Эти стандарты служат основой для создания веб-приложений и сайтов, которые могут использовать все пользователи, включая людей с ограниченными возможностями. Однако соблюдение этих требований не всегда является простым процессом. Часто организации сталкиваются с различными барьерами — от недостатка знаний до отсутствия ресурсов.

В первой части главы мы подробно рассмотрим основные принципы цифровой доступности, включая восприятие, управление, понимание и совместимость. Мы также обсудим законодательные инициативы и нормативные акты, которые поддерживают эти принципы на уровне государственной политики. Затем рассмотрим конкретные рекомендации и лучшие практики для организаций, стремящихся улучшить свою цифровую доступность. Эти рекомендации будут охватывать различные аспекты — от проектирования интерфейсов до тестирования и оценки доступности.

Также необходимо учитывать важность постоянного обучения и адаптации в условиях быстро меняющегося цифрового ландшафта. Цифровая доступность — это не конечная цель, а непрерывный процесс, требующий внимания и усилий со стороны всех участников. Только совместными усилиями мы сможем создать более инклюзивное цифровое пространство для всех пользователей.

### *2.1 Стандарты доступности*

Важным шагом к созданию доступной среды является следование установленным стандартам и рекомендациям. Одним из наиболее известных документов является WCAG, который предлагает набор принципов и критериев для разработки доступного веб-контента.

На основании данного документа во многих странах были выпущены стандарты соответствия доступности:

- Americans with Disabilities Act Standards for Accessible Design (США),
- Accessibility 508 Standards (США),
- ГОСТ Р 52872-2019 (Россия),
- A11y for Ontarians with Dis. Act (Канада),
- EN 301 549 (Европа),
- Disability Discrimination Act 1992 (Австралия)

Каждый из упомянутых стандартов требует создания информационного контента с соблюдением определённых правил, которые могут иметь различный уровень контроля. Тем не менее указанные стандарты в той или иной степени основываются на стандарте WCAG, разработанном Всемирным Веб Консорциумом.

*2.1.1 Стандарт доступности WCAG*

WCAG — это набор рекомендаций, разработанных для повышения доступности веб-контента для людей с различными ограничениями, включая нарушения зрения, слуха, двигательных функций и когнитивные расстройства. Эти рекомендации были созданы Всемирной организацией по стандартизации (W3C) и являются частью инициативы Web Accessibility Initiative (WAI).

*2.1.1.1 Основные принципы WCAG*

WCAG основывается на четырех ключевых принципах, известный как POUR:

1. *Воспринимаемость (Perceivable)*: Информация и пользовательский интерфейс должны быть представлены так, чтобы их можно было воспринять. Например, текст должен быть доступен через альтернативные форматы, такие как аудио или Брайлевские выводы. *Тот кто не видит должен услышать, тот кто не слышит должен увидеть.*

2. *Управляемость (Operable)*: Пользовательский интерфейс и навигация должны быть управляемыми. Это означает, что пользователи должны иметь возможность взаимодействовать с элементами интерфейса, используя клавиатуру или другие вспомогательные технологии. *В программе не должно быть ни одного такого действия, которое не смог бы выполнить пользователь.*

3. *Понятность (Understandable)*: Информация и операции должны быть понятными. Пользователи должны легко понимать, как использовать интерфейс и какие действия они могут предпринять. *Отображаемая информация и способы выполнения действий в программе должны быть интуитивно поняты.*

4. *Надежность (Robust)*: Контент должен быть надежным и совместимым с различными технологиями, включая вспомогательные

средства. Это обеспечивает доступность контента на разных устройствах и платформах. *Передаваемая информация должна быть пригодна для интерпретации большим количеством вспомогательных средств.*

*2.1.1.2 Уровни соответствия*

WCAG определяет три уровня соответствия:

- Уровень A: Минимальные требования, которые должны быть выполнены.
- Уровень AA: Более строгие требования, которые обеспечивают улучшенную доступность.
- Уровень AAA: Наивысший уровень требований, предлагающий максимальную доступность.

*2.1.1.3 Примеры рекомендаций*

Некоторые примеры конкретных рекомендаций из WCAG включают:

- Использование альтернативного текста для изображений.
- Обеспечение достаточного контраста между текстом и фоном.
- Возможность навигации по сайту с помощью клавиатуры.
- Предоставление четких инструкций для заполнения форм.

Следование этим рекомендациям позволяет разработчикам создавать приложения, которые могут использовать все категории пользователей, минимизируя барьеры и улучшая общее качество взаимодействия.

*2.1.1.4 Версии WCAG*

На данный момент выпущены следующие версии WCAG:

1. WCAG 1.0 - опубликована в 1999 году. Это первая версия рекомендаций по доступности веб-контента.

2. WCAG 2.0 - опубликована в 2008 году. Эта версия значительно улучшила и расширила рекомендации по доступности в сравнении с первой

версией. Содержит рекомендации, которые считаются основой для соблюдения доступности.

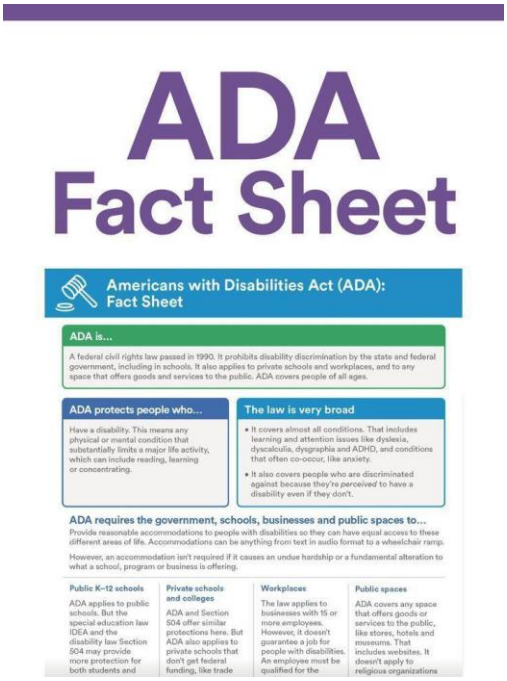
3. WCAG 2.1 - выпущена в 2018 году. Эта версия расширяет предыдущие рекомендации, добавляя новые критерии для улучшения поддержки пользователей с ограниченными возможностями, включая лиц с нарушениями мобильности и когнитивными расстройствами.

4. WCAG 2.2 – принята в октябре 2023 года. В новой редакции Руководства был исключен 1 устаревший Критерий успеха, добавлены 9 новых Критериев и 5 статей в Глоссарий.

Новые Критерии ориентированы в первую очередь на людей с ограниченной моторикой и нарушением когнитивных функций, однако следование им поможет сделать веб-контент более доступным широкому кругу интернет-пользователей, включая пожилых людей, людей с низким уровнем образования или людей с временными ограничениями подвижности рук вследствие травмы.

*2.1.2 Стандарт доступности ADAAG*

Руководство по обеспечению доступности ADA (ADAAG) — это свод стандартов, разработанных в соответствии с Законом об американцах с ограниченными возможностями (ADA – Americans with Disabilities Act). Закон об американцах с ограниченными возможностями (ADA) и его Руководство по доступности для архитектурных объектов (ADAAG – ADA Accessibility Guidelines) определяют требования к доступной среде для обеспечения равного доступа для людей с ограниченными возможностями.




Основные аспекты ADAAG включают:

- Архитектурные требования: Правила, касающиеся конструкции зданий, чтобы они были доступны для всех, включая людей с ограниченными возможностями передвижения.
- Технические критерии: Определяют конкретные размеры и характеристики.
- Охватывает не только физический доступ, но и доступ к другим ресурсам, таким как информационные технологии и услуги.

Информационно-технологическая доступность включают в себя технические нормы для доступности веб-сайтов, программного обеспечения и других технологий, которые помогают людям с ограниченными возможностями получать информацию и участвовать в онлайн-активностях, например, с использованием Брайлевских клавиатур и TTY-устройств.

Требования к информационно-технологической доступности, предусмотренные в ADA (ADAAG), направлены на обеспечение равного доступа к информационным и коммуникационным технологиям для людей с ограниченными возможностями.

Основные положения включают в себя:

1. Доступ к веб-сайтам: Все веб-ресурсы должны соответствовать стандартам доступности, таким как WCAG, чтобы обеспечить легкость навигации, читабельность и использование различных устройств.

2. Аудиовизуальные материалы: Видеоматериалы и мультимедиа должны содержать альтернативные текстовые описания, субтитры и аудиодорожки для поддержки людей с нарушениями слуха и зрения.

3. Интерфейсы пользователей: Интерактивные элементы, такие как кнопки и ссылки, должны быть легкими для понимания и использования с помощью клавиатуры или других вспомогательных технологий.

4. Программное обеспечение: Программные приложения должны быть разработаны с учётом принципов доступности, включая совместимость с программами чтения с экрана и другими вспомогательными устройствами.

5. Общение: Системы связи и обслуживания клиентов должны быть доступны для всех, включая возможность получения информации через альтернативные каналы, такие как текстовые сообщения или видео связи с использованием жестового языка.

6. Обучающие материалы: Учебные материалы и программы должны быть адаптированы для использования людьми с разными типами ограничений, включая использование вспомогательных технологий.

7. Оповещения и уведомления: Все системы оповещения должны быть доступны и учитывать различные способы передачи информации, чтобы обеспечить понимание и реагирование со стороны пользователей.

8. Доступные терминалы: Физические терминалы, такой как автоматизированные банкоматы и киоски, должны быть доступны для людей с ограниченной подвижностью и обеспечивать тактильную навигацию для людей с нарушениями зрения.

9. Поддержка пользователей: Должны предоставляться обучающие материалы и техническая поддержка, чтобы помочь пользователям с ограниченными возможностями освоить информационно-технологические системы.

ADAAG не основывается напрямую на конкретной версии WCAG, однако для обеспечения доступности веб-контента и информационных технологий часто используются рекомендации из WCAG. Наиболее

актуальные принципы и критерии доступности, с которыми часто сопоставляют ADAAG, исходят из WCAG 2.0 и 2.1.

Хотя ADAAG не освещает конкретные версии WCAG, важно, чтобы организации использовали эти веб-стандарты для достижения соответствия требованиям доступности в соответствии с ADA.

*2.1.3 Accessibility 508 Standards*


Стандарты доступности (Accessibility 508 Standards, ADA 508) представляют собой набор требований, направленных на обеспечение доступности информационных технологий для людей с ограниченными возможностями. Эти стандарты были развиты в соответствии с Законом о реабилитации 1973 года и актуализированы на протяжении времени.

Основные требования к доступной среде, которые должны быть соблюдены в соответствии со стандартами 508:

1. Обеспечение доступности: Все электронные и информационные технологии, включая веб-сайты, программы и мультимедиа, должны быть доступны для людей с различными ограничениями, включая физические, сенсорные и когнитивные.

2. Текстовая альтернатива: Вся информация, передаваемая в форме не текстовых средств (например, изображения, графика и видео), должна иметь текстовые альтернативы, которые обеспечивают равнозначный доступ.

3. Навигация с клавиатуры: Все интерфейсы и элементы управления должны быть доступны с помощью клавиатуры, без необходимости использования мыши.

4. Явное обозначение элементов интерфейса: Все элементы интерфейса, такие как кнопки, ссылки и формы, должны быть четко обозначены, чтобы пользователи могли понимать их функции.

5. Адаптивность контента: Информация должна быть представлена так, чтобы ее можно было адаптировать и изменять в зависимости от потребностей пользователя (например, изменение размера шрифта или контрастности).

6. Мультимедиа: Видеоматериалы должны сопровождаться субтитрами или текстовыми транскрипциями, а аудиоматериалы должны иметь текстовые версии для людей с нарушениями слуха.

7. Устойчивость к ошибкам: Пользователи должны иметь возможность легко исправлять ошибки, возникшие в процессе ввода данных или навигации.

8. Мобильные устройства: Все требования доступности должны распространяться на мобильные приложения и сайты, чтобы обеспечить доступность информации на различных устройствах.

9. Проверка доступности: Необходимо регулярно проводить аудит и тестирование на доступность, чтобы идентифицировать и исправлять проблемы.

При разработке стандартов доступности за основу был взят актуальный на тот момент документ WCAG 2.0.

*2.1.4 ГОСТ Р 52872-2019*

Полное название: ГОСТ Р 52872-2019 ИНТЕРНЕТ-РЕСУРСЫ И ДРУГАЯ ИНФОРМАЦИЯ, ПРЕДСТАВЛЕННАЯ В ЭЛЕКТРОННО-ЦИФРОВОЙ ФОРМЕ. ПРИЛОЖЕНИЯ ДЛЯ СТАЦИОНАРНЫХ И МОБИЛЬНЫХ УСТРОЙСТВ, ИНЫЕ ПОЛЬЗОВАТЕЛЬСКИЕ ИНТЕРФЕЙСЫ. Требования доступности для людей с инвалидностью и других лиц с ограничениями жизнедеятельности.


ФЕДЕРАЛЬНОЕ АГЕНТСТВО
ПО ТЕХНИЧЕСКОМУ РЕГУЛИРОВАНИЮ И МЕТРОЛОГИИ

НАЦИОНАЛЬНЫЙ СТАНДАРТ РОССИЙСКОЙ ФЕДЕРАЦИИ

ГОСТ Р 52872— 2019

ИНТЕРНЕТ-РЕСУРСЫ И ДРУГАЯ ИНФОРМАЦИЯ, ПРЕДСТАВЛЕННАЯ В ЭЛЕКТРОННО-ЦИФРОВОЙ ФОРМЕ. ПРИЛОЖЕНИЯ ДЛЯ СТАЦИОНАРНЫХ И МОБИЛЬНЫХ УСТРОЙСТВ, ИНЫЕ ПОЛЬЗОВАТЕЛЬСКИЕ ИНТЕРФЕЙСЫ

Требования доступности для людей с инвалидностью и других лиц с ограничениями жизнедеятельности

Издание официальное

Москва Стандартинформ 2019

Данный стандарт определяет основные требования к созданию доступной среды для лиц с ограниченными возможностями здоровья. Он включает в себя следующие ключевые аспекты:

1. Универсальность проектирования: Все объекты и услуги должны быть доступны для использования всеми людьми, независимо от их физических возможностей.

2. Безбарьерный доступ: Обеспечение свободного доступа к помещениям для людей с ограничениями по движению, включая создание пандусов, лифтов и подходящих лестниц.

3. Информационная доступность: Информация должна предоставляться в различных форматах, включая визуальные и аудиовизуальные, чтобы удовлетворить потребности различных групп пользователей.

4. Элементы навигации: Необходимость установки ясных и понятных указателей, которые помогут людям ориентироваться в пространстве.

5. Доступность коммуникаций: Обеспечение доступных средств связи, таких как слуховые аппараты и другие вспомогательные устройства, позволяющие людям с нарушениями слуха и речи общаться и получать информацию.

6. Физическое оборудование: Наличие вспомогательных объектов, которые соответствуют стандартам доступности.

7. Требования к обслуживающему персоналу: Обучение персонала, работающего в учреждениях и организациях, в вопросах взаимодействия с людьми с ограниченными возможностями.

В рамках данной книги наиболее интересным является информационная доступность.

В соответствии с ГОСТ: «Требования стандарта распространяются на доступность человеко-ориентированных интерфейсов информационных ресурсов и программного обеспечения на стационарных и переносных компьютерах, планшетах, мобильных устройствах, а также на иных устройствах чтения, ввода, просмотра, воспроизведения информации в электронно-цифровой форме. Требования настоящего стандарта относятся не только к ресурсам, размещенным в глобальной сети Интернет, но и к электронно-цифровой информации, распространяемой в сетях передачи данных предприятий, организаций и сообществ, пользователями которых могут оказаться люди с инвалидностью или люди преклонного возраста.»

При разработке этого стандарта за основу был взят актуальный на тот момент документ WCAG 2.1.

*2.1.5 A11y for Ontarians with Dis. Act*

Как было отмечено ранее, A11Y — это нимероним, который расшифровывается как «доступность», поэтому полная расшифровка закона: Accessibility for Ontarians with Disabilities Act (AODA).


Рисунок 6 – На пути к доступному Онтарио

AODA — это канадский закон, который устанавливает стандарты доступности для людей с ограниченными возможностями. Он даёт им возможность полноценно участвовать во всех аспектах повседневной жизни.

AODA распространяется на все организации, зарегистрированные в Онтарио, и устанавливает стандарты доступности в пяти областях: обслуживание клиентов, занятость, информация и коммуникации, транспорт и общественные места.

Согласно закону, публичный веб-контент, созданный после 2012 года (включая сайты, приложения и цифровые документы), должен соответствовать техническим требованиям WCAG 2.0.

Также AODA требует, чтобы все организации, ведущие бизнес в Онтарио, к 2021 году сделали свои сайты и приложения доступными.

*2.1.6 EN 301 549*

EN 301 549 — европейский стандарт, который устанавливает требования к доступности продуктов и услуг информационных и коммуникационных

технологий (ИКТ). К ним относятся сайты, программное обеспечение и цифровые устройства.

Стандарт был опубликован в 2014 году Европейским институтом стандартов телекоммуникаций (ETSI) по запросу Европейской комиссии. В ноябре 2019 года была выпущена обновлённая версия стандарта — 3.1.1.

EN 301 549 используется в государственных закупках, так как важно, чтобы государственные услуги были доступны для всех. Стандарт применим к большинству организаций в Европе.

Последняя версия стандарта, EN 301 549 v 3.2.1, включает полный текст WCAG 2.1 AA.

*2.1.7 Disability Discrimination Act 1992*


Disability Discrimination Act 1992 (DDA)— закон, принятый Парламентом Австралии для защиты от дискриминации по причине инвалидности.

Документ запрещает дискриминацию в сфере занятости, образования, доступа к общественным помещениям, предоставления товаров и услуг, проживания, в клубах и ассоциациях и в других контекстах.

Закон распространяется на людей с временными и постоянными физическими, интеллектуальными, сенсорными, неврологическими, обучающими и психосоциальными расстройствами, заболеваниями или физическими увечьями. Он также охватывает расстройства, которые были у человека в прошлом, и возможные будущие, а ещё расстройства, которые могут появиться в будущем.

Жалобы, поданные на основании закона, рассматривает Австралийская комиссия по правам человека. Австралийская комиссия по правам человека, которая обеспечивает цифровую доступность, рекомендовала выполнять критерии WCAG на уровне соответствия AA.

## *2.2 Требования к цифровой доступности*

### *2.2.1 Критерии успеха WCAG 2.2*

Критерии успеха WCAG 2.2 представляют собой набор рекомендаций, направленных на обеспечение доступности веб-контента для всех пользователей, включая людей с ограниченными возможностями. Данные критерии успеха являются основой методики автора;

Эти критерии делятся на четыре основные категории, каждая из которых соответствует одному из принципов доступности: восприятие, управляемость, понятность и надежность.

В первую очередь, критерии, относящиеся к восприятию, акцентируют внимание на том, как пользователи воспринимают информацию. Например, все не текстовые элементы, такие как изображения и видео, должны иметь текстовые альтернативы. Это позволяет людям с нарушениями зрения использовать вспомогательные технологии для получения информации. Кроме того, важно обеспечить достаточный контраст между текстом и фоном, чтобы текст был легко читаем для пользователей с нарушениями зрения. Также контент должен быть адаптируемым, что означает, что он должен корректно отображаться на различных устройствах и экранах.

Следующий аспект — управляемость. Здесь акцент делается на том, что пользователи должны иметь возможность легко взаимодействовать с элементами интерфейса. Все функции должны быть доступны с клавиатуры, что особенно важно для людей с ограниченной подвижностью. Элементы управления должны иметь четкий визуальный фокус, чтобы пользователи могли видеть, где они находятся в интерфейсе. При изменении фокуса важно избегать неожиданных изменений контента, так как это может сбивать с толку пользователей.

Критерии успеха, связанные с понятностью, подчеркивают важность ясности и простоты контента. Текст должен быть легким для чтения и понимания, что включает в себя использование понятного языка и структурирование информации. Язык контента должен быть четко определен, чтобы пользователи знали, на каком языке они читают информацию. Элементы управления должны работать предсказуемо; например, если кнопка используется для отправки формы, ее функция должна оставаться неизменной. В случае ошибок пользователи должны получать четкие и понятные сообщения о том, как исправить ситуацию.

Наконец, надежность касается совместимости контента с различными браузерами и вспомогательными технологиями. Это означает, что веб-контент должен быть разработан с использованием стандартов веб-разметки и технологий, чтобы обеспечить его корректное отображение и функционирование на разных платформах. Также важно использовать семантическую разметку для упрощения навигации по сайту.

В дополнение к этим основным критериям WCAG 2.2 вводит новые требования, которые направлены на улучшение доступности для пользователей с когнитивными нарушениями и упрощение форм для повышения их удобства использования. Все эти критерии совместно создают более инклюзивный веб-пространство, доступное для всех пользователей вне зависимости от их возможностей.

### *2.3 Рекомендации по выполнению требований цифровой доступности*

Действия по достижению доступности условно можно разделить на несколько уровней.

**Первый уровень – принципы**

Необходимо добиться соответствия основным принципам WCAG.

**Второй уровень – гайдлайны**

Каждый принцип доступности, представленный на первом уровне, конкретизируется через гайдлайны — это определенные рекомендации, которые описывают, каким образом должен быть организован контент для соответствия этому принципу. Например, принцип «Понятность» включает в себя такие гайдлайны, как «Удобочитаемость», «Предсказуемость» и «Помощь при вводе» и другие.

**Третий уровень – критерии оценивания**

Каждый гайдлайн, в свою очередь, раскладывается на критерии оценивания — конкретные механизмы работы интерфейса и контента.

WCAG предоставляет три уровня соответствия доступности: A, AA, AAA. Соответственно, набор критериев под каждый уровень свой.

**Четвертый уровень: достаточные и консультативные техники**

Техники представляют собой рекомендации о том, как действовать для достижения необходимого уровня соответствия. Их использование не является обязательным, но они могут оказать помощь в соблюдении различных принципов доступности.

В соответствии с представленными уровнями можно рекомендовать следующие действия:

**Проведите аудит доступности**: Проведите аудит цифрового контента, чтобы выявить проблемы с доступностью и пробелы в соблюдении рекомендаций по доступности.

**Разработайте политику доступности**: Разработайте политику, которая описывает приверженность вашей организации к доступности и шаги, которые вы предпримете для обеспечения доступности вашего цифрового контента.

**Предоставьте альтернативный текст для изображений**: Все изображения на вашем сайте и в документах должны иметь альтернативные текстовые описания, которые читаются устройствами чтения с экрана.

**Обеспечьте надлежащий цветовой контраст**: Используйте достаточный цветовой контраст, чтобы текст был легко читаем для людей с нарушениями зрения.

**Используйте доступные шрифты**: Используйте шрифты, которые легко читать и понимать людям с нарушениями зрения.

**Предоставление закрытых субтитров и расшифровок для видео**: Предоставлять закрытые субтитры и расшифровки ко всем видеоматериалам, чтобы сделать их доступными для людей с нарушениями слуха.

**Убедитесь, что ваш сайт доступен для клавиатуры**: Убедитесь, что доступ ко всем функциям сайта осуществляется с помощью клавиатуры, не требуя использования мыши или другого указывающего устройства.

**Используйте соответствующую разметку HTML**: Используйте соответствующую разметку HTML, чтобы веб-страницы были правильно структурированы и доступны для чтения с экрана.

**Протестируйте свой цифровой контент с помощью вспомогательных технологий**: Протестируйте свой цифровой контент с помощью программ для чтения с экрана и других вспомогательных технологий, чтобы убедиться, что он доступен для людей с ограниченными возможностями.

## *2.4 Рекомендуемые инструменты тестирования доступности веб-контента*

### *2.4.1 Colour Contrast Analyzer*

В современном мире, где цифровые технологии становятся неотъемлемой частью повседневной жизни, доступность информации для всех пользователей имеет первостепенное значение. Правильный выбор цветового контраста помогает не только улучшить восприятие текста, но и способствует более инклюзивному дизайну. Согласно стандартам WCAG, минимальные требования к контрасту между текстом и фоном определены для обеспечения удобства чтения.

Colour Contrast Analyzer — это специализированный инструмент, предназначенный для оценки контраста между текстом и фоном в цифровых интерфейсах. Он играет важную роль в обеспечении доступности веб-контента для людей с нарушениями зрения, а также для создания визуально привлекательных и читаемых дизайнов.

Данный инструмент позволяет автоматически вычислять уровень контраста между двумя выбранными цветами, которые могут быть заданы в различных форматах, таких как HEX, RGB или HSL. CCA позволяет пользователям выбирать два цвета (например, текст и фон) и автоматически вычисляет коэффициент контрастности между ними. Результаты сравниваются с установленными стандартами, чтобы определить, соответствует ли сочетание цветов требованиям доступности.

CCA показывает, соответствует ли выбранная комбинация цветов требованиям WCAG для различных уровней доступности (AA и AAA). Уровень AA требует минимального контрастного коэффициента 4.5:1 для обычного текста и 3:1 для крупного текста.

CCA может поддерживать разные режимы отображения, такие как «дневной» и «ночной», что полезно для проверки контрастности в разных условиях освещения.

Примеры использования

1. Веб-дизайн: Дизайнеры могут использовать Colour Contrast Analyzer для проверки сочетаний цветов на своих веб-страницах. Например, если они выбирают светло-желтый текст на белом фоне, инструмент покажет, что такой контраст недостаточен для удобного чтения.

2. Разработка приложений: Разработчики мобильных приложений могут интегрировать анализатор в процесс тестирования, чтобы убедиться, что интерфейсы приложений соответствуют стандартам доступности и обеспечивают комфортное взаимодействие для всех пользователей.

3. Реклама и маркетинг: Специалисты по рекламе могут использовать этот инструмент при создании рекламных материалов, чтобы гарантировать, что текст на баннерах и плакатах будет легко читаем даже в условиях плохого освещения или для людей с нарушениями зрения.

4. Образование: Учебные заведения могут применять Colour Contrast Analyzer при разработке учебных материалов и онлайн-платформ, чтобы обеспечить доступность информации для всех студентов, включая тех, кто имеет специфические потребности.

Colour Contrast Analyzer — это незаменимый инструмент в арсенале дизайнеров, разработчиков и маркетологов, стремящихся создать доступный и инклюзивный контент. Он помогает не только соответствовать нормативам, но и заботиться о комфорте пользователей, делая цифровое пространство более доступным.

### *2.4.2 Adobe Acrobat Pro*

Adobe Acrobat Pro — это мощный инструмент, который предоставляет пользователям широкий спектр возможностей для работы с PDF-документами. Одной из ключевых функций этого программного обеспечения является проверка доступности, что особенно важно в контексте создания инклюзивного контента. Доступность PDF-документов позволяет обеспечить равный доступ к информации для людей с различными ограничениями, включая нарушения зрения, слуха и другие.

Adobe Acrobat Pro предлагает встроенный инструмент для проверки доступности, который позволяет быстро оценить соответствие документа стандартам WCAG и PDF/UA (PDF Universal Accessibility).

Он проверяет такие элементы, как текстовые альтернативы для изображений, заголовки, структура документа и контрастность цветов.

Acrobat Pro позволяет создавать PDF-документы с учетом доступности, включая возможность добавления тегов, которые помогают экранным считывателям правильно интерпретировать содержимое.

Acrobat Pro позволяет просматривать и редактировать теги документа, чтобы убедиться, что они правильно отражают структуру и порядок содержания. Теги помогают организовать текст, изображения и другие элементы, делая их более доступными.

Acrobat Pro позволяет добавлять текстовые альтернативы для изображений и графиков, что важно для пользователей экранных считывателей. Также Acrobat Pro предлагает инструменты для проверки их доступности, включая правильное назначение полей и добавление подсказок.

Порядок проверки доступности обычно включает в себя следующие шаги:

- **Запуск проверки:** Для начала пользователю необходимо открыть документ в Adobe Acrobat Pro, затем перейти в меню «Инструменты» и выбрать раздел «Доступность». В этом разделе доступен инструмент «Проверка доступности», который позволяет начать процесс анализа.
- **Анализ документа:** При запуске проверки Acrobat анализирует различные элементы документа, такие как текстовые блоки, изображения, таблицы и ссылки. Он проверяет наличие альтернативного текста для изображений, правильную структуру заголовков, читаемость текста и другие важные аспекты.
- **Результаты проверки:** По завершении анализа Acrobat предоставляет отчет с указанием найденных проблем и рекомендациями по их исправлению. Это позволяет пользователю быстро идентифицировать области, требующие внимания.

После получения отчета о доступности пользователь может приступить к исправлению выявленных проблем. Adobe Acrobat Pro предлагает инструменты для редактирования документа:

- **Добавление альтернативного текста:** Для изображений и графиков можно добавить описательный текст, который будет читаться экранными читалками.
- **Структурирование документа:** Если заголовки не были правильно использованы, их можно изменить или добавить, чтобы обеспечить логическую последовательность информации.
- **Проверка цветового контраста:** Хотя Acrobat не предоставляет прямую проверку цветового контраста, пользователи могут использовать дополнительные инструменты для оценки сочетаний цветов на страницах документа.

Примеры использования Adobe Acrobat Pro для проверки доступности:

1. Создание учебных материалов: Учебные заведения могут использовать Adobe Acrobat Pro для проверки доступности учебных пособий и материалов, чтобы гарантировать, что все студенты, включая тех, кто имеет специфические потребности, могут получить доступ к информации.

2. Подготовка отчетов и презентаций: Компании, готовящие отчеты или презентации в формате PDF, могут проверить свои документы на доступность перед распространением среди клиентов и партнеров. Это создает положительный имидж компании как социально ответственной организации.

3. Публикация научных статей: Исследователи и ученые могут использовать Adobe Acrobat Pro для проверки доступности своих публикаций, чтобы гарантировать, что их работа будет доступна для широкой аудитории, включая людей с ограниченными возможностями.

4. Государственные и муниципальные документы: Государственные учреждения обязаны обеспечивать доступность своих документов для всех граждан. Использование Adobe Acrobat Pro помогает им соответствовать законодательным требованиям и стандартам доступности.

Adobe Acrobat Pro является незаменимым инструментом для всех, кто стремится создать доступный контент. Его функции проверки доступности позволяют пользователям не только выявлять проблемы, но и эффективно их исправлять, обеспечивая тем самым равный доступ к информации для всех. В современном мире, где информация становится все более цифровой, забота о доступности становится важным аспектом в создании инклюзивного общества.

*2.4.3 PAC2021*

PAC (Accessibility Checker) 2021 — это мощный инструмент для проверки доступности документов в формате PDF. Он был разработан с целью

помочь авторам и разработчикам удостовериться, что их контент соответствует международным стандартам доступности, таким как WCAG и PDF/UA (PDF Universal Accessibility).

Инструмент позволяет пользователям выявлять и исправлять потенциальные проблемы доступности, которые могут затруднить взаимодействие с документами для людей с ограниченными возможностями. PAC 2021 предоставляет подробные отчеты о состоянии доступности, а также рекомендации по улучшению.

PAC2021 применяется в различных сферах, где важна доступность информации для всех пользователей, включая:

1. Веб-разработка: Разработчики используют PAC2021 для анализа веб-сайтов и приложений, выявляя элементы, которые могут затруднить доступность для людей с ограничениями по зрению, слуху или моторике.

2. Образование: Учебные заведения применяют инструмент для проверки доступности учебных материалов и платформ, обеспечивая равные возможности для всех студентов.

3. Государственные организации: В соответствии с законодательством о доступности, государственные учреждения используют PAC2021 для проверки своих сайтов и услуг на соответствие требованиям доступности.

PAC2021 функционирует по принципу автоматизированного анализа PDF файлов. Процесс включает несколько ключевых этапов:

1. Сканирование страницы: Пользователь загружает файл PDF. Инструмент сканирует содержимое на наличие элементов, которые могут быть недоступны.

2. Анализ элементов: PAC2021 проверяет различные аспекты страницы, такие как:

- Альтернативный текст для изображений.

- Структура заголовков.
- Контрастность цветов.

3. Отчет о доступности: По завершении анализа инструмент генерирует отчет, в котором перечислены найденные проблемы и рекомендации по их исправлению. Отчет может включать как критические ошибки, так и менее серьезные замечания.

Варианты использования PAC2021:

1. Проверка корпоративного сайта: Компания X разрабатывает новый корпоративный сайт. Перед запуском они используют PAC2021 для проверки доступности всех страниц, выявляя отсутствие альтернативного текста для некоторых изображений и недостаточную контрастность текста. В результате они вносят необходимые изменения, обеспечивая доступность сайта для всех пользователей.

2. Разработка мобильного приложения: Команда разработчиков мобильного приложения использует PAC2021 для проверки интерфейса на доступность. Они получают рекомендации по улучшению структуры и заголовков разделов, что позволяет сделать приложение более инклюзивным.

*2.4.4 NVDA Screen reader*

NVDA (NonVisual Desktop Access) — это бесплатный и открытый считыватель экрана, разработанный для пользователей с нарушениями зрения. Он позволяет людям, использующим клавиатуру и синтезатор речи, взаимодействовать с компьютером, читая текст на экране и предоставляя информацию о графических элементах интерфейса.

NVDA позволяет пользователям перемещаться по различным элементам интерфейса (кнопки, текстовые поля, ссылки и т.д.) с помощью клавиатуры. Пользователь может использовать клавиши со стрелками, Tab и другие комбинации для перехода между элементами.

При фокусировке на элементе NVDA озвучивает его содержимое, предоставляя информацию о тексте, статусе и других атрибутах. Кроме того, NVDA может сообщать о наличии или отсутствии альтернативного текста для изображений. Такой способ позволяет разработчикам понять, как считыватель экрана воспринимает их элементы и что слышат пользователи.

NVDA предоставляет специальные команды для получения информации о заголовках, списках и других структурных элементах. Это помогает разработчикам удостовериться, что их контент правильно структурирован и доступен для навигации.

NVDA поддерживает спецификации ARIA (Accessible Rich Internet Applications), которые помогают разработчикам добавлять доступные атрибуты к элементам интерфейса, что позволяет улучшить взаимодействие с динамическим контентом и сложными интерфейсами.

Разработчики могут использовать NVDA для проверки соответствия стандартам WCAG. Этот процесс включает в себя оценку визуальных элементов, таких как контрастность текста и фона.

Использование NVDA для проверки доступности приложений и веб-сайтов является важным шагом к созданию инклюзивной среды. Этот инструмент помогает разработчикам выявлять проблемы с доступностью и вносить необходимые изменения, чтобы обеспечить комфортное взаимодействие для всех пользователей.

### *2.4.5 JAWS*

JAWS (Job Access With Speech) — это одно из самых популярных программных обеспечений для чтения с экрана, разработанное компанией Freedom Scientific. JAWS — это мощный инструмент для проверки доступности приложений и веб-сайтов. Оно предназначено для людей с

нарушениями зрения и позволяет им взаимодействовать с компьютером и различными приложениями, включая веб-браузеры и офисные программы. Используя JAWS, разработчики могут выявлять проблемы с доступностью и вносить необходимые изменения для улучшения пользовательского опыта. Рассмотрим как работает JAWS и что он предлагает для проверки доступности приложений.

JAWS преобразует текст на экране в речь, позволяя пользователям слышать содержимое документов, веб-страниц и других интерфейсов. Программа позволяет пользователям перемещаться по элементам интерфейса, таким как кнопки, ссылки и формы, используя клавиатурные команды.

JAWS совместим с множеством приложений, включая Microsoft Office, браузеры (Chrome, Firefox, Edge) и другие программы. Кроме основного текста JAWS может озвучивать текстовые альтернативы для изображений, если они были правильно добавлены разработчиками. Для улучшения взаимодействия с конкретными приложениями или веб-сайтами JAWS позволяет создавать и использовать скрипты.

Рассмотрим рекомендации по выполнению тестирования с помощью JAWS:

Тестирование интерфейса:

- Откройте приложение или веб-страницу, которую вы хотите проверить на доступность.
- Используйте клавиатурные команды JAWS для навигации по элементам интерфейса.

Проверка текстовых альтернатив:

- Обратите внимание на изображения и графику; убедитесь, что они имеют текстовые альтернативы (alt-тексты), которые правильно описывают содержание.

Тестирование навигации:

- Проверьте, насколько легко перемещаться по элементам интерфейса (кнопкам, ссылкам и формам) с помощью клавиатуры.
- Убедитесь, что все интерактивные элементы доступны и могут быть активированы без использования мыши.

Анализ результатов:

- Оцените, насколько удобно взаимодействовать с приложением или веб-страницей с помощью JAWS. Обратите внимание на любые проблемы, которые могут затруднять использование.

Хотя JAWS не генерирует автоматические отчеты о доступности, его функции позволяют пользователям вручную оценивать доступность интерфейсов и документировать свои наблюдения. Это может помочь разработчикам улучшить доступность своих приложений.

*2.4.6 VoiceOver*

VoiceOver — это встроенный экранный диктор для устройств Apple, который предоставляет доступ к контенту на экране для пользователей с нарушениями зрения. Он позволяет пользователям взаимодействовать с приложениями и веб-сайтами, используя жесты и клавиатурные команды.

VoiceOver озвучивает текст, кнопки, ссылки и другие элементы интерфейса, когда пользователь перемещает фокус с помощью жестов или клавиш. Пользователи могут перемещаться по элементам интерфейса с помощью жестов (например, свайпов) или клавиатуры. VoiceOver сообщает, какой элемент активен, и его состояние (например, «кнопка включена»).

VoiceOver использует специальные жесты для выполнения различных действий, таких как выбор элемента, прокрутка и активация кнопок. Например, двойной тап на элементе активирует его. Также VoiceOver предоставляет дополнительную информацию о элементах, таких как

подсказки или состояние элементов (например, если переключатель включен или выключен).

Разработчики могут использовать VoiceOver для проверки доступности своего приложения. Это включает в себя навигацию по приложению и оценку, насколько легко взаимодействовать с ним пользователю с нарушениями зрения.

Важно убедиться, что все изображения и графические элементы имеют текстовые альтернативы (alt-тексты), которые VoiceOver может озвучить. Дополнительно VoiceOver помогает проверить правильность структуры заголовков, списков и других элементов, чтобы гарантировать логичное восприятие контента.

### *2.4.7 AXE DevTools*

**AXE DevTools** — это инструмент для проверки доступности веб-приложений и сайтов, разработанный компанией Deque Systems. Он помогает разработчикам и тестировщикам выявлять проблемы доступности и улучшать пользовательский опыт для людей с ограниченными возможностями.

AXE DevTools доступен как расширение для популярных браузеров, таких как Chrome и Firefox. Установив его, вы можете использовать его прямо из инструмента разработчика.

При активации AXE проводит автоматизированный анализ загруженной страницы, проверяя ее на соответствие стандартам доступности, таким как WCAG.

Инструмент генерирует отчет, в котором перечислены найденные проблемы доступности, их серьезность и рекомендации по исправлению. Каждая проблема сопровождается описанием и примерами кода.

AXE позволяет разработчикам взаимодействовать с элементами страницы, чтобы лучше понять, как они могут улучшить доступность. Например, можно просмотреть, какие элементы не соответствуют стандартам. AXE предоставляет возможность тестирования специфических пользовательских сценариев, что позволяет проверить доступность в различных условиях использования.

Инструмент может быть интегрирован в CI/CD процессы, что позволяет командам автоматически проверять доступность при каждом изменении кода.

AXE предлагает обучающие материалы и документацию, что помогает разработчикам лучше понимать принципы доступности и как их применять на практике.

Существуют библиотеки, которые помогают интегрировать AXE в линтеры вашего проекта, например, для React под названием eslint-plugin-jsx-a11y.

*2.4.8 Google Lighthouse*

Google Lighthouse — это инструмент с открытым исходным кодом, который помогает разработчикам улучшать качество веб-страниц. Он предоставляет аудит по различным аспектам, включая производительность, доступность, SEO и лучшие практики. Когда речь идет о тестировании доступной среды, Lighthouse предлагает несколько ключевых функций:

1. Аудит доступности: Lighthouse проверяет веб-страницы на соответствие стандартам доступности, таким как WCAG. Он выявляет проблемы, которые могут затруднить использование сайта людьми с ограниченными возможностями.

2. Отчеты о проблемах: После проведения аудита Lighthouse генерирует отчет, в котором перечислены найденные проблемы и рекомендации по их

исправлению. Например, это могут быть недостаточные контрастные отношения, отсутствие альтернативного текста для изображений или неправильная структура заголовков.

3. Рекомендации по улучшению: Инструмент не только указывает на проблемы, но и предлагает конкретные шаги для их решения. Это может включать добавление ARIA-атрибутов, улучшение навигации с клавиатуры и оптимизацию форм.

4. Интеграция в рабочий процесс: Lighthouse можно интегрировать в CI/CD процессы, что позволяет регулярно проверять доступность на этапе разработки и предотвращать появление новых проблем.

5. Анализ различных устройств: Lighthouse позволяет тестировать страницы на различных устройствах и разрешениях экрана, что помогает убедиться, что доступность не зависит от устройства.

Используя Google Lighthouse для тестирования доступной среды, разработчики могут создать более инклюзивный и удобный для всех пользователей веб-продукт.

*2.4.9 Siteimprove Accessibility Checker For Chrome*

Siteimprove Accessibility Checker для Chrome — это расширение, которое помогает разработчикам и контент-менеджерам проверять доступность веб-страниц непосредственно в браузере. Когда речь идет о тестировании доступной среды, Siteimprove Accessibility Checker предлагает несколько ключевых функций:

1. Установка расширения: Пользователь устанавливает расширение Siteimprove Accessibility Checker из интернет-магазина Chrome.

2. Запуск проверки: После установки пользователь открывает нужную веб-страницу и активирует расширение, кликнув на его иконку в панели инструментов браузера.

3. Автоматический аудит: Расширение проводит автоматическую проверку доступности страницы на соответствие стандартам WCAG. Оно анализирует элементы страницы, такие как изображения, ссылки, заголовки и формы.

4. Отчет о проблемах: После завершения проверки Siteimprove предоставляет отчет, в котором перечислены обнаруженные проблемы с доступностью. Каждая проблема сопровождается описанием и уровнем серьезности (например, критическая, высокая, средняя или низкая).

5. Рекомендации по исправлению: Для каждой проблемы расширение предлагает конкретные рекомендации по исправлению. Это может включать добавление альтернативного текста к изображениям, улучшение контрастности текста и фона, а также оптимизацию навигации.

6. Интерактивный интерфейс: Пользователи могут просматривать проблемы прямо на странице, что облегчает понимание контекста и местоположения каждой ошибки. Это помогает быстрее находить и исправлять недостатки.

7. Сохранение и отслеживание: Расширение позволяет сохранять отчеты и отслеживать изменения в доступности со временем, что особенно полезно для команд, работающих над долгосрочными проектами.

8. Обучение и осведомленность: Использование расширения также способствует повышению осведомленности о важности доступности среди разработчиков и контент-менеджеров, что может привести к более инклюзивным веб-решениям в будущем.

Siteimprove Accessibility Checker является полезным инструментом для обеспечения доступности веб-контента, позволяя выявлять и устранять проблемы на ранних этапах разработки.

*2.4.10 Contrastchecker от WebIM*

Contrastchecker от WebIM — помогает высчитывать коэффициент контрастности. Во WCAG есть критерии, что коэффициент контрастности должен быть не ниже определенного значения. Contrastchecker поможет подобрать цвет, отвечающий всем требованиям.

Contrastchecker от WebIM — это инструмент, предназначенный для проверки контрастности текста и фона на веб-страницах, что является важным аспектом доступности. Вот как он применяется для тестирования доступной среды:

1. Доступ к инструменту: Пользователь открывает сайт Contrastchecker от WebIM в своем браузере.

2. Ввод значений: В интерфейсе инструмента пользователь может ввести значения цветов текста и фона. Это можно сделать с помощью цветовых кодов (например, HEX или RGB) или с помощью цветового выбора.

3. Проверка контрастности: После ввода значений Contrastchecker автоматически вычисляет уровень контрастности между текстом и фоном. Он использует формулы, рекомендованные WCAG, чтобы определить, соответствует ли контрастность установленным стандартам.

4. Результаты проверки: Инструмент предоставляет результаты в виде числового значения (отношение контрастности), а также указывает, соответствует ли это значение минимальным требованиям для различных уровней доступности (AA или AAA).

5. Рекомендации по улучшению: Если контрастность недостаточна, Contrastchecker может предложить альтернативные цветовые комбинации, которые обеспечивают лучший уровень контрастности.

6. Интерактивный интерфейс: Пользователи могут экспериментировать с различными комбинациями цветов, чтобы найти оптимальные решения для повышения доступности их контента.

7. Обучение и осведомленность: Использование Contrastchecker способствует повышению осведомленности о важности контрастности для людей с нарушениями зрения, что может помочь разработчикам и дизайнерам создавать более инклюзивные веб-сайты.

Contrastchecker от WebIM является простым и эффективным инструментом для проверки и улучшения доступности веб-контента, обеспечивая соответствие стандартам контрастности и способствуя созданию более удобных для пользователей интерфейсов.

### *2.5 Возможные технологические решения для обеспечения доступности*

Существует множество технологий, способствующих созданию доступной среды. Одним из самых значительных достижений последних лет стали голосовые помощники и системы распознавания речи. Они позволяют пользователям управлять приложениями без необходимости физического взаимодействия с устройством. Кроме того, использование искусственного интеллекта для адаптации интерфейса под индивидуальные потребности пользователя открывает новые горизонты для доступности.

Можно условно обозначить следующие виды технологических решений для обеспечения доступности:

1. Голосовые интерфейсы и системы распознавания речи

Позволяют людям с нарушениями зрения взаимодействовать с устройствами и приложениями с помощью устной команды. Программы, такие как Siri, Google Assistant и Alexa, демонстрируют, как голосовые

помощники могут облегчить повседневные задачи — от установки напоминаний до управления умным домом.

Системы распознавания речи становятся всё более точными и адаптивными, благодаря чему они могут распознавать акценты и индивидуальные особенности речи. Это делает их доступными для более широкой аудитории, включая людей с речевыми нарушениями. Использование таких технологий не только упрощает взаимодействие с устройствами, но и открывает новые горизонты для пользователей с особыми потребностями.

2. Адаптивные интерфейсы

Адаптивные интерфейсы — это еще одно важное решение для обеспечения доступности. Они способны изменять свой вид и функциональность в зависимости от потребностей пользователя. Например, приложение может предлагать разные варианты отображения текста: увеличить шрифт для людей с нарушениями зрения или изменить цветовую гамму для людей с дальтонизмом.

Технологии машинного обучения и искусственного интеллекта позволяют приложениям запоминать предпочтения пользователей и автоматически подстраиваться под них. Это создает индивидуализированный опыт, который делает взаимодействие более комфортным и интуитивным.

3. Текстовые альтернативы и субтитры

Создание текстовых альтернатив для визуального контента — это важный аспект доступности. Описание изображений, графиков и других визуальных элементов позволяет людям с нарушениями зрения понимать информацию наравне с другими пользователями. Субтитры для видео также играют ключевую роль в обеспечении доступности для людей с нарушениями слуха.

# ГЛАВА 3. ПРОГРАММА ТЕСТИРОВАНИЯ ДОСТУПНОСТИ: ПОДХОДЫ И ПОРЯДОК

В современном мире, где технологии стремительно развиваются и проникают во все сферы жизни, доступность цифровых ресурсов становится неотъемлемой частью их проектирования и разработки. В частности, приложения, которые мы используем ежедневно, должны быть доступны для всех пользователей, независимо от их физических возможностей или ограничений. В этом контексте требования WCAG 2.2 играют ключевую роль в обеспечении инклюзивности веб-контента и приложений.

Тестирование доступности — это не просто техническая процедура, а важный этап в создании пользовательского опыта, который учитывает разнообразие потребностей пользователей. Оно требует внимательного анализа как визуальных, так и функциональных аспектов интерфейса, а также применения различных методик и инструментов для оценки соответствия установленным стандартам.

Тестирование программных продуктов, в том числе, тестированием доступности занимаются инженеры по обеспечению качества (QA), таким специалистом является автор книги.

Инженеры по обеспечению качества (QA) — это специалисты, занимающиеся оценкой и улучшением качества программного обеспечения. Их основная задача заключается в выявлении дефектов и проблем в продукте до его выпуска, чтобы обеспечить соответствие требованиям и ожиданиям пользователей.

При рассмотрении подходов к тестированию затронем такой важный инструмент инженеров по обеспечению качества как контрольные списки (или чек листы) и опишем программу тестирования, разработанную автором для собственной уникальной методики.

Мы также уделим внимание практическим рекомендациям по тестированию, включая использование автоматизированных инструментов и ручных методов, а также важность привлечения пользователей с ограниченными возможностями к процессу тестирования.

### *3.1 Порядок тестирования доступности веб-контента*

Рассмотрим порядок тестирования на примере веб-приложения, предназначенного для онлайн-заказа еды. Тестирование доступности веб-контента выполняется с помощью четырех видов испытаний:

1) Предварительное тестирование на этапе разработки.
2) Функциональное тестирование. Если контент не работает должным образом, он будет недоступен не только людям с ограниченными возможностями, но всем пользователям в принципе.
3) Проверка контента на соответствие стандартам доступности (например, WCAG 2.2).
4) - Убедиться, что приложение удобно для пользователей с ограничениями по зрению и слуху.
5) Юзабилити тестирование доступного контента с учетом разных жизненных ситуаций.

#### *3.1.1 Предварительное тестирование*

| Этап | Подэтап | Действия |
|---|---|---|
| Подготовка | Определение целей тестирования | Установите, какие аспекты доступности важны для вашего приложения (например, для пользователей с ограниченными возможностями по зрению) |
| | Сбор требований | Изучите стандарты доступности, такие как WCAG |
| Выбор инструментов и ресурсов | Выбор инструментов | Определите инструменты для проверки доступности (например, Adobe Acrobat Pro). |
| | Тестовые платформы | Подготовьте различные устройства и платформы (мобильные и десктопные) для тестирования |

| Этап | Подэтап | Действия |
|---|---|---|
| | Создание контрольных списков | Подготовьте списки критериев доступности для проверки и список сценариев для их проверки. |
| Автоматизированное тестирование | Запуск инструментов | Используйте автоматизированные инструменты для сканирования веб-приложения на наличие ошибок доступности. |
| | Анализ результатов | Обработайте результаты и составьте отчет о найденных проблемах. |
| Ручное тестирование | Тестирование с использованием экранного чтеца | Проверьте, как приложение воспринимается с помощью экранных чтецов |
| | Навигация с клавиатуры | Убедитесь, что все элементы интерфейса доступны с помощью клавиатуры. |
| | Тестирование контрастности | Проверьте контрастность текста и фона |
| | Тестирование форм | Убедитесь, что все формы имеют правильно указанные метки и подсказки. |
| Повторное тестирование | Проверка исправлений | После внесения изменений выполните повторное тестирование, чтобы убедиться в исправлении ошибок |
| | Регулярное тестирование | Установите регулярные проверки доступности в процессе разработки. |
| Документация и обучение | Документирование результатов | Сохраните результаты тестирования и исправления для будущих итераций |
| | Обучение команды | Проведите обучение для команды разработчиков по вопросам доступности и ее тестирования |

### *3.1.2 Функциональное тестирование*

Функциональное тестирование веб-приложения для людей с ограниченными возможностями включает в себя проверку различных аспектов доступности и удобства использования.

| Этап | Подэтап | Действия |
|---|---|---|
| Подготовка к тестированию | Определение целевой аудитории | Понять, какие именно ограничения могут иметь пользователи (например, слабовидящие, пользователи с двигательными нарушениями). |
| | Выбор инструментов | Использование инструментов для автоматизированного и ручного тестирования. |
| | Составление сценариев тестирования | Определение ключевых функций веб-приложения, которые будут проверяться. |
| Проведение тестирования | Регистрация и вход в систему | - Проверить доступность формы регистрации для экранных читалок.<br>- Убедиться, что все поля имеют соответствующие метки (label) и описания (aria-label).<br>- Проверить возможность навигации по форме с помощью клавиатуры. |

| Этап | Подэтап | Действия |
|---|---|---|
| | Поиск ресторанов | - Убедиться, что поле поиска поддерживает ввод с клавиатуры и голосовых команд.<br>- Проверить доступность списка результатов поиска для экранных читалок. |
| | Выбор блюда | - Проверить, что все блюда имеют четкие описания и изображения.<br>- Убедиться, что кнопки «Добавить в корзину» доступны с помощью клавиатуры и имеют достаточный контраст. |
| | Корзина | - Проверить доступность интерфейса корзины для экранных читалок.<br>- Убедиться, что пользователи могут изменять количество товаров с помощью клавиатуры. |
| | Оформление заказа | - Проверить доступность формы оформления заказа.<br>- Убедиться, что все поля имеют правильные метки и подсказки.<br>- Проверить возможность завершения заказа с использованием только клавиатуры. |
| | Обратная связь | - Убедиться, что пользователи могут оставить отзыв о заказе.<br>- Проверить доступность формы обратной связи. |
| Инструменты и методы | Автоматизированное тестирование | - Проанализировать собранные данные, выделяя основные проблемы и области для улучшения.<br>- Обратить внимание на частые ошибки и трудности. |
| | Ручное тестирование | - Оценить приложение по стандартам доступности (например, WCAG).<br>- Сравнить результаты с ожиданиями и целями тестирования. |
| Документация результатов | Составление отчета о тестировании | Составление отчета о выявленных проблемах, включая описание проблемы, шаги для воспроизведения и рекомендации по исправлению. |
| | Приоритезация проблем | Классификация проблем по уровню критичности (высокий, средний, низкий). |
| Ретестинг | Ретестинг | После внесения исправлений необходимо провести повторное тестирование для проверки устранения выявленных проблем. |

Автоматизированные инструменты. Для автоматизированного функционального тестирования веб-приложения по доставке еды можно использовать различные инструменты. Вот некоторые из них:

**1. Selenium**

Один из самых популярных инструментов для автоматизации браузеров. Позволяет писать тесты на различных языках программирования (Java, C#, Python и др.).

Применение: Можно использовать для автоматизации сценариев, таких как регистрация, поиск ресторанов, добавление блюд в корзину и оформление заказа.

**2. Cypress**

Современный инструмент для тестирования веб-приложений с простым синтаксисом и возможностью запуска тестов в реальном времени.

Применение: Идеален для тестирования интерфейсов, позволяет легко писать и отлаживать тесты.

**3. TestCafe**

Инструмент для автоматизированного тестирования, который не требует установки плагинов и работает с любым браузером.

Применение: Подходит для тестирования веб-приложений, включая мобильные версии.

**4. Puppeteer**

Описание: Библиотека Node.js для управления браузером Chrome или Chromium. Позволяет выполнять автоматизированные действия в браузере.

Применение: Можно использовать для создания скриншотов, генерации PDF-файлов и выполнения функциональных тестов.

**5. Playwright**

Новый инструмент от Microsoft, который поддерживает несколько браузеров (Chromium, Firefox, WebKit).

Применение: Подходит для написания кросс-браузерных тестов и может использоваться для тестирования сложных сценариев.

**6. Postman**

Инструмент для работы с API, который позволяет тестировать и документировать API запросы.

Применение: Полезен для тестирования бэкенда приложения, например, проверки корректности работы API для получения списка ресторанов или оформления заказа.

**7. Jest**

JavaScript-тестовый фреймворк, который часто используется вместе с React-приложениями.

Применение: Может использоваться для юнит-тестирования компонентов интерфейса и логики приложения.

**8. Katalon Studio**

Полнофункциональная платформа для автоматизации тестирования, которая поддерживает как веб-, так и мобильные приложения.

Применение: Удобна для создания и управления тестами без необходимости глубоких знаний программирования.

**10. Robot Framework**

Универсальный фреймворк для автоматизации тестирования с использованием ключевых слов.

Применение: Можно использовать для функционального тестирования веб-приложений с помощью библиотек Selenium или Requests.

Эти инструменты помогут автоматизировать процесс тестирования веб-приложения по доставке еды, повысив его эффективность и качество.

Ручное тестирование. Рассмотрим пример программы для ручного тестирования веб-приложения для заказа еды. Она включает в себя основные сценарии тестирования и шаги, которые необходимо выполнить.

1. Тестирование регистрации пользователя

Цель: Проверить, что новый пользователь может успешно зарегистрироваться.

Шаги:

1) Открыть главную страницу приложения.
2) Нажать на кнопку «Регистрация».
3) Ввести валидные данные (имя, email, пароль).
4) Нажать «Зарегистрироваться».

Ожидаемый результат: Пользователь должен быть успешно зарегистрирован и перенаправлен на страницу профиля.

2. Тестирование входа пользователя

Цель: Убедиться, что зарегистрированный пользователь может войти в систему.

Шаги:

1) Открыть главную страницу приложения.
2) Нажать на кнопку «Вход».
3) Ввести email и пароль зарегистрированного пользователя.
4) Нажать «Войти».

Ожидаемый результат: Пользователь должен быть успешно авторизован и перенаправлен на главную страницу с доступом к меню.

3. Тестирование поиска ресторанов

Цель: Проверить, что пользователь может искать рестораны по названию или категории.

Шаги:

1) Войти в систему.
2) Ввести название ресторана или категорию в строку поиска.
3) Нажать «Поиск».

Ожидаемый результат: На экране должны отобразиться соответствующие результаты поиска.

4. Тестирование добавления товара в корзину

Цель: Убедиться, что пользователь может добавить блюда в корзину.

Шаги:

1) Найти ресторан и открыть его меню.
2) Выбрать блюдо и нажать «Добавить в корзину».
3) Повторить для нескольких блюд.

Ожидаемый результат: Все выбранные блюда должны отображаться в корзине.

5. Тестирование оформления заказа

Цель: Проверить, что пользователь может оформить заказ.

Шаги:

1) Перейти в корзину.
2) Проверить список товаров.
3) Нажать «Оформить заказ».
4) Ввести адрес доставки и выбрать способ оплаты.
5) Нажать «Подтвердить заказ».

Ожидаемый результат: Заказ должен быть успешно оформлен, пользователь должен увидеть сообщение о подтверждении заказа.

6. Тестирование истории заказов

Цель: Убедиться, что пользователь может просмотреть историю своих заказов.

Шаги:

1) Войти в систему.
2) Перейти на страницу «Мой профиль» или «История заказов».

Ожидаемый результат: Должен отображаться список всех предыдущих заказов с их статусами.

7. Тестирование выхода из системы

Цель: Проверить, что пользователь может выйти из системы.

Шаги:

1) Нажать на кнопку «Выход» в меню профиля.

Ожидаемый результат: Пользователь должен быть успешно разлогинен и перенаправлен на страницу авторизации.

Эта программа охватывает основные функциональные аспекты веб-приложения для заказа еды и может быть дополнена другими тестами в зависимости от специфики приложения и требований бизнеса. Рекомендуется также вести учет найденных ошибок и результатов тестирования для последующего анализа.

Функциональное тестирование веб-приложений для людей с ограниченными возможностями требует тщательной подготовки и применения различных методов. Важно учитывать потребности пользователей на каждом этапе разработки и тестирования, чтобы создать инклюзивный продукт.

*3.1.3 Проверка контента на соответствие требованиям WCAG*

После функционального тестирования можно переходить к проверке контента на соответствие требованиям WCAG. Рассмотрим пример программы проверки контента веб-приложения для заказа еды на соответствие требованиям WCAG.

По методике автора основное внимание должно уделяться проверке семантики разметки HTML, так как правильная семантическая разметка HTML является ключевым аспектом создания доступных веб-приложений для пользователей

с ограниченными возможностями. Данные проверки включают в себя следующие проверки:

- Использование семантических тегов
- Заголовки и структура
- Альтернативный текст для изображений
- Формы и их элементы
- Ссылки и кнопки
- Таблицы
- Использование ARIA (если необходимо)

В связи с этим, программа включает в себя основные аспекты доступности и шаги для их проверки.

1. Проверка правильной семантической разметки HTML

Семантические теги помогают браузерам и вспомогательным технологиям понять структуру и содержание страницы.

Цель: проверить корректность семантической разметки HTML для отображения в браузере.

Данный шаг включает в себя проверки на использование только следующих семантических тегов HTML разметки:

- <header> — для заголовка страницы или секции.

- <nav> — для навигационных ссылок.

- <main> — для основного содержимого страницы.

- <section> — для разделов контента.

- <article> — для независимого контента (например, блога или новостной статьи).

- <aside> — для вспомогательной информации, которая не является основной (например, боковая панель).

- <footer> — для нижнего колонтитула страницы или секции.

Ожидаемый результат: Семантическая разметка HTML должна содержать только регламентированные семантические теги.

2. Проверка правильной структуры и заголовков HTML разметки

Цель: проверить корректность структуры и используемых заголовков HTML для отображения в браузере.

Данный шаг включает в себя следующие проверки заголовков и логической структуры HTML документа:

- Используются заголовки (<h1>, <h2>, <h3>, и т.д.) для создания иерархии контента.
- Страница начинается с одного <h1>, который описывает основную тему.
- Используются более низкие уровни заголовков для подзаголовков, чтобы создать логическую структуру.

Ожидаемый результат: Логическая структура и используемые теги не нарушают установленных правил.

3. Проверка альтернативного текста для изображений

Цель: Убедиться, что все изображения имеют соответствующий альтернативный текст.

Шаги:

1) Просмотреть все изображения на страницах приложения.
2) Проверить наличие атрибута <alt> у изображений и убедиться, что текст описывает содержание изображения.

*<img src="image.jpg" alt="Описание изображения">*

Ожидаемый результат: Все изображения должны иметь описательный альтернативный текст.

4. Проверка форм и элементов управления

Цель: Убедиться, что формы доступны для пользователей с ограниченными возможностями.

Шаги:

1) Проверить, что каждое поле формы имеет соответствующий ярлык (label), связанный с соответствующим полем через атрибут for.

```
<label for="username">Имя пользователя:</label>
<input type="text" id="username" name="username">
```

2) Убедиться, что ошибки валидации форм отображаются ясно и доступны для понимания.

Ожидаемый результат: Все поля формы должны быть правильно помечены и доступными для пользователей с экранными читалками.

5. Проверка ссылок и кнопок

Цель: Убедиться, что используются семантически правильные элементы для интерактивных элементов.

Шаги:

1) Проверить, что используются теги <a> для ссылок, а теги <button> для кнопок.

```
<a href="https://example.com">Перейти на сайт</a>
<button type="button">Отправить</button>
```

2) Проверить, что теги  или  с JavaScript не используются для создания кнопок.
3) Проверить, что теги  или  с JavaScript не используются для создания кнопок.

Ожидаемый результат: Для интерактивных элементов должны использоваться семантически правильные элементы.

6. Проверка таблиц

Цель: Убедиться, что используются правильные семантические элементы таблиц.

Шаги:

1) Проверить, что для структурирования таблиц используются теги <table>, <thead>, <tbody>, и <tfoot>.
2) Проверить, что заголовки столбцов указаны с помощью <th> и используется атрибут scope для указания области действия заголовка.

```
<table>
  <thead>
    <tr>
      <th scope="col">Имя</th>
      <th scope="col">Возраст</th>
    </tr>
  </thead>
  <tbody>
    <tr>
      <td>Иван</td>
      <td>30</td>
    </tr>
  </tbody>
</table>
```

Ожидаемый результат: Для таблиц должны использоваться используются правильные семантические элементы.

7. Проверка использования ARIA-атрибутов

Если семантическая разметка не может полностью описать элемент, необходимо использовать ARIA-атрибуты, чтобы улучшить доступность,

например, с использованием *role, aria-label*, и другие атрибуты, чтобы предоставить дополнительную информацию.

Роли определяют, как элемент будет восприниматься вспомогательными технологиями.

Основные роли (ARIA Roles):

- role="button" — кнопка.
- role="checkbox" — чекбокс.
- role="dialog" — диалоговое окно.
- role="grid" — сетка.
- role="link" — ссылка.
- role="list" — список.
- role="navigation" — навигационное меню.
- role="tabpanel" — панель вкладки.
- role="slider" — ползунок.

Свойства (ARIA Properties) предоставляют дополнительную информацию о состоянии или характеристиках элемента.

Примеры свойств:

- aria-label — текстовая метка для элемента.
- aria-labelledby — ссылается на другой элемент, который содержит текстовую метку.
- aria-describedby — указывает на элемент, который описывает текущий элемент.
- aria-hidden — указывает, скрыт ли элемент от вспомогательных технологий (true или false).
- aria-required — указывает, является ли поле обязательным (true или false).

Состояния (ARIA States) отражают текущее состояние элемента.

Примеры состояний:

- aria-checked — состояние чекбокса или переключателя (true, false, или mixed).

- aria-expanded — указывает, развернут ли элемент (например, выпадающее меню).

- aria-selected — указывает, выбран ли элемент в списке или группе элементов.

- aria-disabled — указывает, отключен ли элемент (true или false).

ARIA Relationships (Связи)

Эти атрибуты помогают установить связи между элементами.

- Примеры:

- aria-controls — указывает на элемент, который контролируется (например, кнопка и соответствующее меню).

- aria-flowto — указывает порядок фокуса между элементами.

- aria-haspopup — указывает, что элемент вызывает всплывающее меню или диалог.

Цель: Убедиться, что ARIA-атрибуты используются правильно для улучшения доступности.

Шаги:

1) Проверить использование атрибутов ARIA (например, aria-label, aria-labelledby, aria-hidden) в интерактивных элементах.

Ожидаемый результат: ARIA-атрибуты должны использоваться корректно для улучшения доступности интерфейса.

8. Проверка текстового контента

Цель: Убедиться, что текстовая информация доступна для пользователей с ограниченными возможностями.

Шаги:

1) Проверить, что текст имеет достаточный контраст с фоном (минимум 4.5:1 для обычного текста и 3:1 для большого текста).
2) Убедиться, что размер шрифта достаточен для чтения (рекомендуется не менее 16px).
3) Проверить наличие заголовков и подзаголовков, чтобы структура текста была ясной.

Ожидаемый результат: Все текстовые элементы должны быть читабельными и хорошо структурированными.

9. Проверка навигации с клавиатуры

Цель: Убедиться, что все элементы интерфейса доступны через клавиатуру.

Шаги:

1) Использовать клавишу Tab для перемещения между интерактивными элементами (кнопки, ссылки, поля ввода).
2) Проверить, что все элементы могут быть активированы с помощью клавиатуры.

Ожидаемый результат: Пользователь должен иметь возможность полностью взаимодействовать с приложением без использования мыши.

10. Проверка мультимедийного контента

Цель: Убедиться, что мультимедийный контент доступен для всех пользователей.

Шаги:

1) Проверить наличие субтитров и транскрипций для видео и аудио контента.

Ожидаемый результат: Все мультимедийные материалы должны иметь доступные альтернативы.

11.Проверка адаптивности дизайна

Цель: Убедиться, что приложение адаптируется к различным устройствам и экранам.

Шаги:

1) Проверить отображение приложения на различных устройствах (мобильные телефоны, планшеты, десктопы).

Ожидаемый результат: Интерфейс должен быть удобным и доступным на всех устройствах.

Эта программа охватывает проверку основных аспектов доступности веб-приложения в соответствии с требованиями WCAG. Рекомендуется проводить данные проверки как при разработке, так и при тестировании готового программного продукта.

*3.1.4 Юзабилити тестирование*

Тестирование с пользователями: Привлечение людей с ограниченными возможностями для тестирования приложения и получения обратной связи.

Рассмотрим примерный порядок юзабилити тестирования приложения для заказа еды с привлечением людей с ограниченными возможностями.

Таблица 1 – Порядок тестирования с пользователем

| **Этап** | **Подэтап** | **Действия** |
|---|---|---|
| Подготовка | Определение целей тестирования | Определить, какие аспекты доступности будут оцениваться (например, навигация, читаемость, взаимодействие). |
| | Составление сценариев | Создать сценарии использования приложения, отражающие реальные задачи (например, оформление заказа, поиск ресторана) |
| | Подбор участников | - Найти людей с различными ограничениями (слабовидящие, люди с нарушениями слуха, моторными нарушениями и т.д.).<br>- Убедиться, что участники имеют разнообразный опыт использования технологий. |
| Проведение тестирования | Предварительный инструктаж | - Объяснить участникам цель тестирования и его процесс.<br>- Убедиться в комфорте участников и их готовности. |
| | Наблюдение | - Попросить участников выполнить подготовленные сценарии.<br>- Наблюдать за их действиями, фиксируя трудности и успехи. |

| Этап | Подэтап | Действия |
|---|---|---|
| | Сбор данных | - Записывать комментарии участников, их реакции и поведение.<br>- Использовать видеозапись (с согласия участников) для последующего анализа. |
| Анализ результатов | Обработка данных | - Проанализировать собранные данные, выделяя основные проблемы и области для улучшения.<br>- Обратить внимание на частые ошибки и трудности. |
| | Оценка доступности | - Оценить приложение по стандартам доступности (например, WCAG).<br>- Сравнить результаты с ожиданиями и целями тестирования. |
| Рекомендации и улучшения | Составление отчета | - Подготовить отчет с описанием выявленных проблем и рекомендациями по их устранению.<br>- Включить примеры из тестирования для наглядности. |
| | Презентация результатов | - Представить результаты заинтересованным сторонам (разработчикам, дизайнерам и менеджерам). |
| Повторное тестирование | Внедрение изменений | - После внесения изменений в приложение провести повторное тестирование с теми же или новыми участниками. |
| | Оценка улучшений | - Сравнить результаты повторного тестирования с предыдущими для оценки эффективности внесенных изменений. |
| Итоговая оценка | Обратная связь | - Собрать обратную связь от участников о том, как изменения повлияли на их опыт использования приложения. |
| | Долгосрочные планы | - Разработать стратегию для регулярного тестирования доступности в будущем. |

Этот порядок поможет выявить проблемы доступности и улучшить пользовательский опыт для людей с ограниченными возможностями.

### *3.2 Контрольные списки: Новый подход к тестированию программного обеспечения*

Роль инженеров по обеспечению качества (QA) и их потребности быстро растут в связи с быстро меняющимися сценариями разработки программного обеспечения. Исторические тенденции, связанные с процессом тестирования программного обеспечения, подразумевают использование некоторых традиционных подходов, таких как контрольные списки, тестовые случаи и тестовые сценарии для определения качества программного продукта.

Традиции действительно важнее всего, но поскольку технологии меняются, методологии тестирования должны меняться.

В этой книге рассмотрим традиционный подход и предложим альтернативу: «Test Accessibility». Test Accessibility - это инновационный подход, представленный автором книги - QA-инженером и экспертом в области тестирования доступности программного обеспечения.

Для начала давайте окунемся в традиционные методы организации и управления тестами. Эти традиционные методологии являются строительными лесами, поддерживающими систематическую и структурированную экспертизу программного приложения. Вышеупомянутые проверенные временем методологии определяют рамки, в которых QA-инженеры могут перемещаться по лабиринтам тестирования таким образом, чтобы тщательно протестировать критические функциональные возможности.

Эти контрольные списки содержат перечень критериев, тестовые примеры - подробные пошаговые инструкции по проверке, а тестовые сценарии - сквозные оценки - правильно определенную стратегию тестирования.

Это важно знать, потому что они закладывают основу того, что новые инновационные подходы обещают поднять эффективность тестирования программного обеспечения на более высокий уровень в нашей постоянно меняющейся технологической экосистеме.

Контрольные списки являются базовыми для использования в тестировании программного обеспечения, так как они дают возможность убедиться, что важнейшие шаги выполняются в хронологическом порядке, не пропуская ни одного. Контрольные списки легко разрабатывать, они просты по структуре, но им не хватает нескольких деталей: в них отсутствует структура, указывающая последовательность элементов или задач для

проверки, или в них не указано, как должна быть выполнена задача или каков должен быть результат ее выполнения.

Контрольные списки очень полезны, когда нужно ускорить процесс быстрого и неформального тестирования, но они недостаточно эффективны для достижения уровня полноты и глубины, характерного для систематического тестирования. Такие списки проверяемых элементов или задач являются очень полезным инструментом, в основном когда необходимо быстро оценить элементарные функциональные возможности.

Тем не менее, по своей природе они коротки и довольно просты, что позволяет им охватить лишь узкий круг многих сложных сценариев и исключений, которые подбрасывают сложные программные приложения.

Полагаясь на контрольные списки, вы рискуете упустить момент в динамичной и постоянно развивающейся среде разработки, поскольку они недостаточно гибкие, чтобы удовлетворить изменившиеся требования или изменения, возникающие из-за непредвиденных проблем.

Комплексное и системное тестирование требует гораздо более широкого подхода, включающего множество различных методов тестирования, работающих вместе, чтобы представить полное исследование функциональности, надежности и общего качества тестируемого программного обеспечения.

Хотя контрольные списки занимают важное место в инструментарии тестирования, их роль должна быть дополнена более мощными инструментами, такими как тест-кейсы и тестовые сценарии, которые помогут повысить эффективность процесса тестирования и создать более мощное программное обеспечение.

Тест-кейсы гораздо более конкретны и предоставляют структуру того, как каждая функция или функциональность должна быть оценена в

отдельности. Наиболее распространенная структура состоит из следующего: ID тестового случая, название тестового случая, предварительные условия, шаги, ожидаемые результаты, фактические результаты, прохождение/непрохождение, тестовая среда и комментарии.

Будучи столь конкретными и подробными, они имеют компромисс: они требуют строгости и повторения при их поддержании и создании. Более того, написание этих пошаговых инструкций требует понимания тонкостей программного обеспечения на очень детальном уровне, на что QA-инженерам приходится тратить много времени.

И эта задача становится еще более сложной по мере совершенствования программного обеспечения, когда в его существующий код вносятся новые возможности, функциональность и изменения.

Такая динамичность опасна тем, что тест-кейсы по своей природе легче всего устаревают и теряют актуальность в результате дальнейших усилий по разработке. В результате организации сталкиваются с потенциально десятками тысяч устаревших тест-кейсов.

Но если обеспечить точное соответствие тест-кейсов определенным требованиям и спецификациям, то их эффективность будет прямо пропорциональна тому, насколько они соответствуют изменяющейся динамике программного обеспечения и насколько проактивны усилия по сопровождению.

Преимущество специфичности тестовых примеров в рамках тестирования программного обеспечения должно быть сбалансировано с проблемой их трудозатратности и потенциального устаревания.

Из этого можно сделать вывод, что тест-кейсы имеют структуру повествования, которая отражает всесторонние действия пользователя с программным обеспечением. Тест-кейс начинается с высокого уровня,

краткого описания некоторой истории пользователя или случая использования, давая фоновый контекст, в котором требуется оценить функциональность программного обеспечения.

Предварительное условие органично сочетается с условиями, которые должны быть заданы заранее и должны быть обеспечены или удовлетворены до разработки сценария; то есть предварительное условие задает отправную точку, подобно фундаменту, чтобы обеспечить последовательное начало моделирования взаимодействия с пользователем.

Из описания видно, что детали использования тестовых данных в сценарии будут включать в себя все: входные значения, конфигурации и переменные, определяющие реакцию, которую программное обеспечение должно обеспечить во время выполнения. Интеграция соответствующих тестовых данных, таким образом, сделает сценарий более реалистичным и обеспечит достоверную модель поведения пользователя.

Но недостатком может стать тот факт, что рамки тестового сценария носят описательный характер. Их высокоуровневая природа делает их очень удобными для отображения реалистичных пользовательских взаимодействий. Однако эта же характеристика может приводить к расплывчатости и двусмысленности. При такой гибкости, присущей сценариям, каждая деталь должна быть задокументирована и доведена до сведения команды.

Затем необходимо разработать рекомендации, учитывающие возможные вариации интерпретации.

Чтобы избежать недостатков традиционных подходов к тестированию программного обеспечения, автор книги предлагает инновационный способ: «Тестовые наброски». Тестовые наброски - это вид тестовой документации, который может унаследовать положительные стороны контрольных списков,

тестовых случаев и тестовых сценариев, но при этом свободен от их недостатков.

Идея Test Outlines - это переосмысление традиционного подхода, присутствующего в тестовых кейсах, и просто - новый подход, который вводит повествование, подобное тому, что присутствует в связности и контексте тестовых сценариев.

Такое сочетание методологий закладывает основу для подхода к тестированию, который является более дальновидным, чем его предшественники. Повествовательная структура Test Outlines выходит за рамки всех шагов тестового сценария и вместо этого выстраивает эти шаги в убедительную сюжетную линию путешествия пользователя по программному обеспечению.

Это позволяет не только упростить общую документацию по тестированию, но и создать целостное представление о том, как конечные пользователи будут взаимодействовать с программным обеспечением в реальных условиях.

Глубина позволяет гораздо шире понимать процесс тестирования, превращая его из простого контрольного списка шагов в динамическую эвристику, ориентированную на пользовательский опыт. С другой стороны, описательный подход позволяет перейти от изолированной функциональности к взаимосвязям между функциями.

Это позволяет выявить критические зависимости, потенциальные проблемы интеграции и поведение системы в целом во время работы с пользовательским интерфейсом. Повествование - это инструмент, облегчающий когнитивные функции при мысленном моделировании пользовательского опыта и его вероятных проблем со стороны команд тестирования.

Тестовый конспект, по сути, представляет собой смену парадигмы: уплотнение обычных пошаговых тестовых примеров для повышения эффективности процесса тестирования за счет повествовательного исследования.

Это новый подход, который преодолевает ограничения традиционных методологий и, по сути, лучше понимает тонкую динамику, установленную между программным обеспечением и его пользователями, что позволяет проводить более эффективное и всестороннее тестирование.

# ГЛАВА 4. МЕТОДИКА - ОСНОВА ХОРОШЕГО ТЕСТА. МЕТОДИКА ТЕСТИРОВАНИЯ ЭЛЕМЕНТОВ ДОСТУПНОЙ СРЕДЫ В ПРИЛОЖЕНИЯХ

Настоящая глава посвящена методике тестирования элементов доступной среды в приложениях с учетом требований WCAG 2.2, которая была разработана автором книги. Мы рассмотрим основные принципы и подходы, которые помогут разработчикам и тестировщикам выявить и устранить барьеры, мешающие пользователям с ограниченными возможностями полноценно взаимодействовать с цифровыми продуктами.

В рамках решения вопросы методики тестирования мы обсудим ключевые критерии успеха WCAG 2.2, их влияние на проектирование интерфейсов и методы их проверки, а также рассмотрим методику тестирования доступной среды для веб-приложения, включающую себя тестирование на этапе разработки и тестирование готового программного продукта.

Данная глава станет основой для понимания того, как обеспечить доступность приложений, способствуя созданию более инклюзивного цифрового пространства, где каждый пользователь сможет найти нужную информацию и эффективно взаимодействовать с функционалом приложения.

## *4.1 Методика тестирования доступной среды на этапе разработки*

Как было замечено в предыдущем разделе, самое лучшее – начинать тестирование программного продукта на самом раннем этапе – на этапе разработки, так как у потенциальных пользователей программного продукта бывают очень разные ограничения, например, туннельное зрение, разное цветовое восприятие, приступы как эпилепсия из-за невернойне верной анимации (яркое мигание света, дергание картинки) дислексия (подробнее

этот вопрос был рассмотрен в разделе 1.2), а все потенциальные ограничения пользователей необходимо учитывать еще на этапе разработки.

*4.1.1 Установка аспектов доступности*

Аспекты доступности

1. Визуальная доступность

Визуальная доступность касается людей с нарушениями зрения. Важными аспектами здесь являются контрастность цветов, возможность увеличения шрифта и использование альтернативного текста для изображений. Например, текст на сайте должен быть четким и контрастным, чтобы его могли читать люди с частичной потерей зрения. Альтернативный текст позволяет пользователям экранных читалок понимать содержание изображений.

2. Аудиальная доступность

Аудиальная доступность важна для людей с нарушениями слуха. Это включает в себя наличие субтитров для видео и текстовых транскрипций для аудиоматериалов. Субтитры не только помогают людям с полной потерей слуха, но и могут быть полезны для людей, которые не владеют языком, на котором ведется речь в видео.

3. Кинестетическая доступность

Кинестетическая доступность ориентирована на пользователей с физическими ограничениями. Важно учитывать возможность навигации по веб-ресурсам с помощью клавиатуры, а не только мыши. Это особенно актуально для людей с ограниченной подвижностью, которые могут использовать специальные устройства для ввода информации.

4. Когнитивная доступность

Когнитивная доступность касается пользователей с умственными нарушениями или дислексией. Простота и ясность интерфейса являются

ключевыми аспектами. Использование простого языка, четких инструкций и логичной структуры помогает всем пользователям легче воспринимать информацию.

Практические шаги по установке аспектов доступности

1. Оценка текущего состояния

Первым шагом к улучшению доступности является оценка текущего состояния веб-ресурса или приложения. Это можно сделать с помощью различных инструментов и методик, которые позволяют выявить существующие проблемы.

2. Обучение команды

Важно обучить разработчиков и дизайнеров основам доступности. Это поможет им создавать более инклюзивные продукты и учитывать потребности разных пользователей на всех этапах разработки.

3. Внедрение стандартов

Следует придерживаться международных стандартов доступности, таких как WCAG. Эти рекомендации помогают разработчикам создавать контент, который будет доступен для всех категорий пользователей.

4. Тестирование на разных устройствах

Тестирование на различных устройствах и с использованием различных технологий вспомогательных средств позволяет убедиться в том, что контент действительно доступен для всех пользователей.

*4.1.2 Определение элементов приложения, требующих проверку на доступность*

Доступность в контексте программного обеспечения означает, что приложение должно быть пригодным для использования людьми с различными ограничениями, такими как нарушения зрения, слуха или моторики. Это подразумевает наличие функций, которые помогают

пользователям взаимодействовать с приложением без барьеров. Ключевые аспекты доступности включают:

- Семантическая разметка: Использование правильных HTML-тегов для обеспечения корректного восприятия контента вспомогательными технологиями.

- Контрастность: Обеспечение достаточной разницы между цветами текста и фона для удобства чтения.

- Навигация с клавиатуры: Возможность полного взаимодействия с приложением без использования мыши.

- Проверка ARIA-меток: Все интерактивные элементы должны иметь правильные ARIA-метки и описания.

- Проверка изображений: Все изображения должны иметь атрибуты alt. Проверьте, что текст альтернативы описывает содержание изображения.

- Проверка форм: Все поля форм должны иметь метки и подсказки, доступные для экранных читалок.

Доступность в контексте цифровых приложений подразумевает создание условий, при которых пользователи с различными ограничениями могут взаимодействовать с контентом и функциями приложения без каких-либо барьеров. Это включает в себя людей с нарушениями зрения, слуха, двигательными недостатками и когнитивными расстройствами. Основная цель проверки доступности — выявить и устранить потенциальные проблемы, которые могут помешать пользователям эффективно использовать приложение.

Для более детального анализа доступности приложения необходимо классифицировать его элементы. Основные категории включают:

Для более детального анализа доступности приложения необходимо классифицировать его элементы. Основные категории включают:

1. Визуальные элементы

Визуальные элементы приложения, такие как текст, изображения и графические интерфейсы, требуют особого внимания. Ключевые аспекты, которые следует учитывать:

- Контрастность: Цвета текста и фона должны иметь достаточный контраст для удобства чтения. Это особенно важно для пользователей с нарушениями зрения.

- Размер шрифта: Шрифты должны быть достаточно крупными и легко читаемыми. Возможность изменения размера текста без потери функциональности также является важным аспектом.

- Альтернативный текст: Все изображения должны содержать альтернативный текст, который описывает их содержание. Это позволяет пользователям экранных читалок понимать контекст изображения.

2. Аудио и видео контент

Аудиовизуальные материалы должны быть доступны для пользователей с нарушениями слуха и зрения. Основные элементы проверки:

- Субтитры: Видео должно иметь субтитры, чтобы люди с нарушениями слуха могли понять содержание.

- Транскрипция: Аудиоматериалы должны сопровождаться текстовыми транскрипциями, что позволит пользователям читать содержание вместо прослушивания.

- Описание видео: Для пользователей с нарушениями зрения необходимо предоставлять описания ключевых визуальных элементов.

3. Элементы навигации

Навигация является критически важной частью любого приложения. Элементы навигации должны быть интуитивно понятными и доступными для всех пользователей:

- Кнопки и ссылки: Они должны быть четко обозначены и легко нажимаемы. Размер кнопок должен быть достаточным для удобного взаимодействия.

- Клавиатурная навигация: Приложение должно поддерживать навигацию с помощью клавиатуры, что особенно важно для пользователей с ограниченной подвижностью.

- Структура меню: Логическая и последовательная структура меню помогает пользователям быстро находить нужные разделы.

4. Формы и ввод данных

Формы являются важной частью взаимодействия пользователя с приложением. Их элементы также требуют проверки на доступность:

- Ясные метки: Каждое поле ввода должно иметь четкую метку, объясняющую его назначение.

- Ошибки ввода: В случае ошибок в заполнении формы пользователи должны получать понятные сообщения об ошибках и рекомендации по исправлению.

- Поддержка вспомогательных технологий: Формы должны быть совместимы с экранными читалками и другими вспомогательными средствами.

Процесс проверки доступности

Для эффективной проверки элементов приложения на доступность можно использовать следующие шаги:

1. Аудит доступности

Первым этапом является проведение аудита существующего приложения с использованием специализированных инструментов и методик. Это позволит выявить проблемные области и составить план действий.

2. Пользовательское тестирование

Важно вовлекать пользователей с ограниченными возможностями в процесс тестирования. Их отзывы помогут понять реальные проблемы и улучшить интерфейс.

3. Обучение команды

Обучение разработчиков и дизайнеров принципам доступности поможет создать более инклюзивные продукты в будущем. Это включает изучение стандартов WCAG и лучших практик.

4. Постоянный мониторинг

Доступность — это не разовая задача, а постоянный процесс. Регулярный мониторинг и обновление приложения помогут поддерживать его на высоком уровне доступности.

*4.1.3 Определение инструментов, выполняющих проверку на доступность*

Для обеспечения доступности необходимо использовать различные инструменты, позволяющие проводить проверку на соответствие стандартам. Рассмотрим несколько ключевых инструментов авторской методики, которые помогают разработчикам и дизайнерам выполнять проверку доступности.

PAC 2021 (Проверка доступности контента) — это мощный инструмент для анализа веб-контента на предмет его доступности. Он предоставляет пользователям возможность загружать веб-страницы и проверять их на соответствие стандартам WCAG. PAC 2021 анализирует элементы страницы, такие как текст, изображения, ссылки и формы, и выявляет потенциальные проблемы.

Одной из главных особенностей PAC 2021 является возможность предоставления подробных отчетов с рекомендациями по исправлению найденных ошибок. Это делает его полезным не только для тестирования, но и для обучения разработчиков принципам доступности. Инструмент также

поддерживает различные языки, что делает его доступным для международной аудитории.

Adobe Acrobat Pro — это инструмент, который часто используется для работы с PDF-документами. Однако его возможности не ограничиваются лишь редактированием файлов. Он также предоставляет функции проверки доступности документов. С помощью Adobe Acrobat Pro пользователи могут проверять наличие альтернативного текста для изображений, правильного структурирования заголовков и других элементов, необходимых для обеспечения доступности.

Инструмент предлагает автоматизированный анализ, а также возможность вручную проверять каждую часть документа. Результаты проверки отображаются в виде отчетов, что позволяет пользователям легко идентифицировать проблемные области и вносить необходимые изменения.

NVDA (NonVisual Desktop Access) — это бесплатный экранный читалка для операционной системы Windows. Он позволяет пользователям с нарушениями зрения взаимодействовать с компьютерными приложениями и веб-сайтами. NVDA поддерживает различные языки и предоставляет возможность настройки голосового вывода.

Использование NVDA для проверки доступности веб-приложений позволяет разработчикам понять, как их интерфейс воспринимается людьми с ограниченными возможностями. Инструмент помогает выявить проблемы с навигацией, отсутствием альтернативного текста.

JAWS (Job Access With Speech) — это один из самых популярных экранных читалок, используемых людьми с нарушениями зрения. Он предлагает широкий спектр функций, включая поддержку различных приложений и веб-браузеров. JAWS позволяет пользователям получать информацию о содержимом экрана в аудиоформате.

Для разработчиков JAWS является важным инструментом для тестирования доступности. Используя эту читалку, проверяется, как приложения воспринимаются пользователями с нарушениями зрения, а также выявить проблемы с навигацией и взаимодействием с интерфейсом.

AXE DevTool — это расширение для браузеров Chrome и Firefox, которое позволяет разработчикам быстро проверять веб-страницы на наличие проблем с доступностью. Оно предоставляет автоматизированные тесты на соответствие стандартам WCAG и предлагает рекомендации по исправлению найденных ошибок.

AXE DevTool имеет удобный интерфейс и позволяет пользователям проводить тестирование в реальном времени. Это делает его идеальным инструментом для интеграции в процесс разработки, позволяя выявлять проблемы на ранних этапах.

Lighthouse — это инструмент с открытым исходным кодом от Google, который позволяет проводить аудит веб-приложений на предмет производительности, SEO и доступности. Lighthouse предоставляет подробные отчеты о состоянии доступности сайта и предлагает рекомендации по улучшению.

Инструмент может быть использован как в виде расширения для браузера Chrome, так и через командную строку. Lighthouse анализирует различные аспекты доступности, включая использование ARIA-атрибутов, наличие альтернативного текста и структурирование контента.

Использование инструментов для проверки доступности является необходимым шагом в процессе разработки веб-приложений. Каждый из рассмотренных инструментов — PAC 2021, Adobe Acrobat Pro, NVDA, JAWS, VoiceOver, AXE DevTool и Lighthouse — играет важную роль в обеспечении инклюзивного пользовательского опыта. Они помогают разработчикам

выявлять проблемы на разных этапах создания продукта и предоставляют рекомендации по их устранению. В конечном счете, использование этих инструментов способствует созданию более доступных и удобных приложений для всех пользователей.

*4.1.4 Выполнение тестирования*

После того, как мы определились с инструментами, необходимо рассмотреть методику их применения. Порядок проверок приведен в разделе 3.1.3. Приступать к тестированию необходимо на самом раннем этапе разработки, когда только выполнена разметка страниц.

1. Проверка правильной семантической разметки HTML осуществляется: 1.1. выполняем конвертацию HTML документа в pdf с сохранением всех тэгов.
    1.2. проверяем .pdf файл на теги с помощьюпомощь PAC2021 (или более новой версии) Adobe Acrobat Pro (именно PRO версия).
    1.3. если есть логотипы и/или ссылки, то выполняется проверка, что использованы верные тэги.
2. Проверка правильной логической структуры и заголовков HTML с помощью файла .pdf (для создания иерархии контента используются заголовки <h1>, <h2>, <h3>, и т.д.; страница начинается с одного <h1>, который описывает основную тему).
3. Проверка файла .pdf на наличие альтернативного текста для изображений (наличие тегов <alt>).
4. Проверка файла .pdf на то, что каждое поле формы имеет соответствующий ярлык (label), связанный с соответствующим полем через атрибут for.
5. Проверка файла .pdf на то, что используются теги <a> для ссылок, а теги <button> для кнопок.

6. Проверка файла .pdf на то, что используемые правильные семантические элементы таблиц

   6.1. проверяем, что для структурирования таблиц используются теги <table>, <thead>, <tbody>, и <tfoot>

   6.2. проверяем, что заголовки столбцов указаны с помощью <th> и используется атрибут scope
7. Проверка использования атрибутов ARIA (например, aria-label, aria-labelledby, aria-hidden) в интерактивных элементах.
8. С помощью Colour Contrast Analyzer пПроверяем какой шрифт, какой размер шрифта, цвет и цвет фона, чтобы все было максимально читабельно. Для слабовидящих или тех, кто совсем не видит, используется NVDA Screen reader, JAWS и VoiceOver.
9. Проверка навигации с клавиатуры

   9.1. отключаем мышь и используем только клавиши Tab, Shift + Tab и Enter для навигации.

   9.2. проверяем, можно ли получить доступ ко всем элементам интерфейса.

   9.3. убеждаемся, что порядок навигации логичен и последовательный.

   10. Проверка мультимедийного контента на наличие субтитров и транскрипций для видео и аудио контента.

   11. Проверка адаптивности дизайна при использовании приложения на различных устройствах (мобильные телефоны, планшеты, компьютеры, ноутбуки).

Резюме: Генерирование PDF файла из HTML позволит нам тестировать сам HTML код на раннем этапе, и уже правильный код конвертировать в HTML страницу. Проверка PDF включает в себя:

1. Присутствует и распознается header (шапка документа).
2. ТегТэк заголовка (H1 tag) – только один
3. Ссылки содержат link тегтэг
4. Лого содержит logo тегтэг
5. Присутствует и распознается footer (нижняя часть)
6. Верно идентифицированы таблицы (шапка таблицы, колонки, ячейки).

Далее после обратной конвертации из PDF в HTML с помощью NVDA Screen reader проверяется, что таблицы распознаются и на HTML странице, а также что выполняется навигация внутри таблицы, NVDA Screen reader читает не только ячейку, но и название колонки и строчки где эта ячейка расположена.

Далее на HTML странице с помощью AXE Dev Tool осуществляются те же проверки, что выполнялись с pdf файлом.

### *4.2 Методика тестирования доступной среды готового программного продукта*

Перед началом тестирования необходимо ознакомиться с документацией на продукт. Это включает в себя технические спецификации, пользовательские руководства и описания интерфейса. Анализ документации позволит выявить потенциальные проблемы, которые могут возникнуть у пользователей с ограниченными возможностями.

Автоматизированное тестирование — это первый шаг в оценке доступности. Существуют различные инструменты, такие как Axe, Lighthouse и WAVE, которые позволяют быстро проверить веб-приложения на наличие распространенных проблем с доступностью. Эти инструменты могут выявлять

ошибки, такие как отсутствие альтернативного текста для изображений, неправильное использование заголовков и другие нарушения стандартов.

После автоматизированного тестирования следует провести ручное тестирование, которое позволяет глубже понять, как пользователи взаимодействуют с продуктом. На этом этапе важно привлекать людей с ограниченными возможностями для проведения тестов. Они могут предоставить ценные отзывы о том, насколько удобно им использовать интерфейс, какие элементы вызывают трудности и что можно улучшить.

- Тестирование с экранным читалкой: Использование экранных читалок (например, NVDA или JAWS) позволяет проверить, насколько доступен контент для пользователей с нарушениями зрения.

- Тестирование клавиатурной навигации: Проверка возможности навигации по интерфейсу с помощью клавиатуры без использования мыши. Это важно для пользователей с двигательными ограничениями.

- Оценка контрастности: Проверка цветового контраста между текстом и фоном для обеспечения читаемости.

- Проверка ARIA-меток: Проверка, что все интерактивные элементы имеют правильные ARIA-метки и описания.

После завершения тестирования необходимо проанализировать полученные результаты. Важно выделить основные проблемы, которые были выявлены в ходе тестирования, и составить список рекомендаций по их устранению. Также стоит оценить, насколько продукт соответствует установленным стандартам доступности.

Тестирование доступной среды — это итеративный процесс. После внесения изменений в продукт следует повторно проводить тестирование, чтобы убедиться в том, что все выявленные проблемы были устранены и не появились новые.

Критерии оценки доступности могут варьироваться в зависимости от типа продукта и его целевой аудитории. Однако можно выделить несколько общих аспектов:

- Навигация: Удобство навигации по интерфейсу как с помощью клавиатуры, так и с использованием вспомогательных технологий.

- Читаемость: Наличие четкого контраста между текстом и фоном, использование понятных шрифтов и размеров текста.

- Альтернативный текст: Наличие альтернативного текста для всех изображений и мультимедийных элементов.

- Структурирование контента: Правильное использование заголовков и списков для упрощения восприятия информации.

Тестирование доступной среды готового программного продукта — это сложный и многогранный процесс, который требует внимания к деталям и понимания потребностей различных групп пользователей. Следуя описанным этапам и используя соответствующие инструменты, разработчики могут создать инклюзивные приложения, которые будут доступны всем пользователям. В конечном итоге это не только улучшает качество продукта, но и способствует созданию более справедливого общества, где технологии служат всем без исключения.

### *4.3 Сравнение автоматизированных и ручных методов тестирования*

Тестирование программного обеспечения является важным этапом в процессе разработки, обеспечивающим качество и надежность конечного продукта. Можно выделить два основных подхода к тестированию: автоматизированное и ручное. Каждый из этих методов имеет свои преимущества и недостатки, которые необходимо учитывать при выборе

подхода к тестированию. В данной работе мы подробно рассмотрим оба метода, их особенности, области применения и влияние на процесс разработки.

Ручное тестирование — это процесс проверки программного обеспечения, выполняемый вручную тестировщиками. Он включает в себя выполнение тестовых сценариев без использования автоматизированных инструментов. Этот метод позволяет тестировщикам интуитивно оценивать функциональность приложения, а также выявлять ошибки, которые могут быть незаметныне заметны при автоматизированном тестировании.

Ручное тестирование — это процесс проверки программного обеспечения, выполняемый вручную тестировщиками. Он включает в себя выполнение тестовых сценариев без использования автоматизированных инструментов. Этот метод позволяет тестировщикам интуитивно оценивать функциональность приложения, а также выявлять ошибки, которые могут быть не заметны при автоматизированном тестировании.

Преимущества ручного тестирования:

1. Гибкость. Ручное тестирование позволяет быстро адаптироваться к изменениям в требованиях или функциональности приложения. Тестировщики могут легко изменять тестовые сценарии в зависимости от текущих условий.

2. Человеческий фактор. Тестировщики могут использовать свой опыт и интуицию для выявления проблем, которые могут быть упущены автоматизированными системами. Это особенно важно при тестировании пользовательского интерфейса и пользовательского опыта.

3. Легкость в освоении. Для начала ручного тестирования не требуется глубоких технических знаний. Это делает его доступным для широкого круга специалистов.

Недостатки ручного тестирования

1. Временные затраты. Ручное тестирование может занимать много времени, особенно при выполнении повторяющихся задач, таких как регрессионное тестирование.

2. Человеческие ошибки. Тестировщики могут допускать ошибки из-за усталости или невнимательности, что может привести к пропуску критических дефектов.

3. Ограниченная повторяемость. Повторение тестов вручную может быть неэффективным и трудоемким, особенно в больших проектах.

Автоматизированное тестирование — это процесс, при котором тесты выполняются с использованием специализированных инструментов и скриптов. Этот метод позволяет автоматизировать выполнение тестовых сценариев, что значительно ускоряет процесс тестирования.

Преимущества автоматизированного тестирования

1. Скорость выполнения. Автоматизированные тесты могут выполняться значительно быстрее, чем ручные, что особенно важно для больших проектов с частыми изменениями.

2. Повторяемость. Автоматизация позволяет легко повторять тесты без риска человеческой ошибки, что повышает надежность результатов.

3. Экономия ресурсов. Хотя первоначальные затраты на разработку автоматизированных тестов могут быть высокими, в долгосрочной перспективе они позволяют сэкономить время и ресурсы на повторном тестировании.

Недостатки автоматизированного тестирования

1. Начальные затраты. Разработка автоматизированных тестов требует значительных временных и финансовых затрат на создание и поддержку скриптов.

2. Ограниченные возможности. Автоматизация может быть сложной для выполнения некоторых типов тестирования, например, для оценки пользовательского интерфейса или пользовательского опыта.

3. Зависимость от инструментов. Эффективность автоматизированного тестирования зависит от выбранных инструментов и их способности справляться с конкретными задачами.

Ручное и автоматизированное тестирование могут использоваться в разных областях в зависимости от специфики проекта. Например, ручное тестирование часто применяется на начальных этапах разработки для проверки новых функций и пользовательского интерфейса. В то же время автоматизированное тестирование становится более актуальным на этапах регрессионного тестирования и производительности, где необходимо проверить большое количество функционала за короткий срок.

На практике лучше всего использовать комбинированный подход, сочетая оба метода для достижения наилучших результатов. Например, ручное тестирование может использоваться для первичной проверки новых функций, а затем автоматизированные тесты могут быть разработаны для последующих проверок и регрессионного тестирования.

# ГЛАВА 5. ПОДГОТОВКА РЕЛИЗА ПРОГРАММНОГО ПРОДУКТА С ЭЛЕМЕНТАМИ ДОСТУПНОЙ СРЕДЫ

В современном мире программное обеспечение становится неотъемлемой частью нашей повседневной жизни. С каждым годом требования пользователей к функциональности и доступности программных продуктов растут, что делает процесс подготовки релиза особенно актуальным. В данной главе мы рассмотрим ключевые аспекты, связанные с подготовкой релиза программного продукта, акцентируя внимание на важности интеграции элементов доступной среды. При подготовке к релизу обязательно должны учитываться программа и методика проверки элементов доступности, которые были рассмотрена в главе 3 и 4 данной книги.

Цель данной главы заключается в том, чтобы предоставить читателю комплексное представление о процессе подготовки релиза программного продукта с учетом принципов доступности. Мы стремимся ответить на следующие вопросы:

1. Какие этапы включает в себя подготовка релиза
2. Как оценить уровень доступности программного продукта
3. Какие материалы необходимо подготовить для пользователя, чтобы ему комфортно было начинать работать с разработанным программным продуктом

Подготовка релиза программного продукта с элементами доступной среды — это не только вопрос технической реализации, но и социальная ответственность разработчиков. В этой главе мы постараемся показать, что доступность может и должна быть интегрирована на всех этапах разработки, начиная с концепции и заканчивая финальным продуктом.

### *5.1 Анализ и исправление*

Предварительное тестирование программного продукта — это важный этап, который позволяет выявить ошибки и недочеты до его официального релиза. Этот процесс не только помогает улучшить качество конечного продукта, но и способствует повышению удовлетворенности пользователей. В данном разделе мы рассмотрим методы анализа и исправления ошибок, выявленных в ходе предварительного тестирования (см. глава 4), а также проиллюстрируем их на примерах.

Анализ ошибок начинается с систематизации информации о выявленных недочетах. На этом этапе важно классифицировать ошибки по нескольким критериям:

**Тип ошибки:** функциональные, производительные, визуальные и т.д.

**Серьезность ошибки в части функционала:** критические, высокие, средние, низкие.

**Какой уровень доступности должен обеспечить элемент или процесс, в котором обнаружена ошибка:** A, AA, AAA**.**

**Область возникновения:** фронтенд, бэкенд, база данных.

Определите, какие ошибки требуют немедленного внимания, а какие могут быть устранены в будущих обновлениях.

Например, в части обеспечения доступности, ошибка с высоким приоритетом —- это отсутствие альтернативного текста для ключевых изображений, ошибка с низким приоритетом - косметические изменения в дизайне. А в части функционала, если в ходе тестирования была обнаружена ошибка в интерфейсе, которая мешает пользователю выполнить основное действие (например, отправить форму), то ее следует отнести к *критическим*. В то время как ошибка, связанная с отображением незначительного элемента дизайна, может быть классифицирована как *низкая*.

В тестировании программного обеспечения часто встречаются различные виды ошибок, и их распространенность может варьироваться в зависимости от проекта, методологии разработки и других факторов. Вот общее представление о трех категориях ошибок:

1. Функциональные ошибки:

Описание: Ошибки, связанные с неправильным выполнением функциональных требований.

Примеры: Неверный расчет, отсутствие возможности регистрации пользователя, ошибки в логике процессов.

Статистика: По данным различных исследований, функциональные ошибки составляют около 60-70% всех ошибок в программном обеспечении.

2. Производительные ошибки:

Описание: Ошибки, влияющие на производительность приложения, такие как медленная загрузка страниц или высокое потребление ресурсов.

Примеры: Долгое время отклика сервера, утечки памяти, проблемы с масштабируемостью.

Статистика: Производительные ошибки могут составлять от 15% до 25% всех ошибок.

3. Визуальные ошибки:

Описание: Ошибки, касающиеся интерфейса пользователя (UI), такие как неправильное отображение элементов или проблемы с адаптивностью.

Примеры: Неправильное выравнивание кнопок, несоответствующий шрифт, несоответствующая контрастность, проблемы с отображением на мобильных устройствах.

Статистика: Визуальные ошибки обычно составляют около 10-15% от общего числа ошибок.

Эти цифры могут варьироваться в зависимости от типа приложения и конкретного контекста разработки. Важно проводить регулярное тестирование и анализ для выявления и устранения всех видов ошибок на разных этапах разработки.

Результаты анализа необходимо свести в таблицу. Пример такой таблицы представлен ниже.

Таблица 2 – Пример таблицы с результатами анализа

| № | Описание ошибки | Тип ошибки | Серьезность ошибки | Уровень доступности | Область возникновения | Инструмент |
|---|---|---|---|---|---|---|
| 1 | Недостаточная контрастность красного цвета | визуальная | высокие | AA | фронтенд | |
| 2 | Отсутствует альтернативный текст для рисунка | функциональные | критические | AA | фронтенд | |
| 3 | Не удается выполнить вход в программу | производительные | критические | - | бэкенд | |
| | … | | | | | |

Обнаруженные ошибки, кроме документации, должны быть зафиксированы в централизованной системе регистрации ошибок, такой как redmine. Это позволит планировать выпуск релиза и хранить историю ошибок и вариантов их устранения.

### *5.2 Повторное тестирование*

Перед началом повторного тестирования необходимо выполнить несколько подготовительных шагов:

- Анализ предыдущих результатов: Изучите результаты предыдущих тестов на доступность, чтобы понять, какие проблемы были выявлены и как они были устранены (см. раздел 5.1).

- Определение критериев успеха: Установите четкие критерии для оценки доступности. Это могут быть как количественные показатели

(например, время на выполнение задач), так и качественные (удовлетворенность пользователей).

- Создание тестовой среды: Убедитесь, что тестовая среда соответствует реальным условиям использования приложения. Это может включать использование различных устройств и программных средств для имитации ограничений пользователей.

### *5.3 Документация и обучение*

Эффективная эксплуатационная документация и обучение пользователей — это ключевые факторы успешного внедрения программного обеспечения. Они помогают пользователям быстро освоиться с продуктом, минимизируют количество ошибок и повышают общую удовлетворенность от работы с программой.

Разработка эксплуатационной документации и обучение работе с программным обеспечением — это важные этапы, которые обеспечивают успешное внедрение и использование продукта. Процесс можно разделить на несколько ключевых блоков:

1. Подготовительный этап

1.1. Определение целей и задач

- Цели: Определить, какую информацию необходимо включить в документацию и обучение.

- Задачи: Установить сроки, ресурсы и ответственных за создание документации.

1.2. Сбор информации

- Интервью с разработчиками: Получение технической информации о функционале программы.

- Анализ пользовательских требований: Изучение потребностей конечных пользователей.

2. Разработка эксплуатационной документации

2.1. Структурирование документации

- Содержание: Определение основных разделов (введение, установка, использование, устранение неисправностей и т.д.).

- Форматирование: Выбор формата (PDF, HTML, Wiki и т.д.) для удобства доступа.

2.2. Написание документации

- Введение: Общее описание программы, ее назначения и возможностей.

- Установка: Пошаговые инструкции по установке ПО.

- Использование: Описание интерфейса, функций и возможностей программы с примерами использования.

- Устранение неисправностей: Часто задаваемые вопросы и решения распространенных проблем.

2.3. Рецензирование и тестирование

- Проверка документации: Вовлечение технических специалистов и пользователей для проверки точности и полноты информации.

- Тестирование инструкций: Проверка пошаговых инструкций на практике.

3. Обучение пользователей

3.1. Разработка обучающих материалов

- Презентации: Создание слайдов с ключевыми моментами работы с программой.

- Видеоуроки: Запись видео с демонстрацией функционала программы.

3.2. Проведение обучающих сессий

- Очные или онлайн-тренинги: Организация занятий для пользователей.

- Практические занятия: Предоставление возможности пользователям самостоятельно поработать с программой под руководством инструктора.

4. Создание доступной среды

4.1. Адаптация материалов

- Учет потребностей пользователей с ограниченными возможностями: Использование доступных форматов (например, текстовые описания к видео).

- Простота языка: Использование понятного языка без излишне технических терминов.

4.2. Обратная связь от пользователей

- Сбор отзывов: Опросы и анкеты для оценки эффективности обучения и качества документации.

- Корректировка материалов: Внесение изменений на основе полученной обратной связи.

5. Поддержка и обновление документации

5.1. Регулярное обновление

- Мониторинг изменений в программе: Обновление документации при изменении функционала.

- Обновление обучающих материалов: Корректировка в соответствии с новыми версиями ПО.

5.2. Поддержка пользователей

- Горячая линия или чат поддержки: Организация канала для оперативной помощи пользователям.

- Дополнительные обучающие сессии: Проведение дополнительных тренингов по мере необходимости.

### *5.4. Подготовка обучающих материалов для пользователей*

Создание обучающих материалов с учетом потребностей людей с нарушениями зрения и слуха — это важный шаг к обеспечению равного доступа к информации. Включение элементов доступной среды не только улучшает опыт пользователей, но и способствует более широкому принятию программы среди различных групп населения.

Подготовка обучающих материалов для пользователей программы с учетом потребностей людей с нарушениями зрения и слуха требует особого подхода и внимательности. Вот основные шаги и рекомендации:

1. Анализ потребностей пользователей

1.1. Исследование аудитории

- Определение типов нарушений: Понимание особенностей пользователей с нарушениями зрения и слуха.

- Сбор информации: Опросы или интервью с потенциальными пользователями для выявления их потребностей.

2. Разработка контента

2.1. Для пользователей с нарушениями зрения

Текстовые материалы:

- Используйте четкий и простой язык.

- Применяйте крупный шрифт и высококонтрастные цветовые схемы для печатных материалов.

Аудио-материалы:

- Записывайте голосовые инструкции, которые могут быть использованы с программами экранного доступа.

- Используйте четкую дикцию и избегайте фонового шума.

Интерактивные элементы:

- Обеспечьте поддержку экранных читалок (например, правильная разметка HTML-документов).

- Используйте альтернативный текст для изображений.

2.2. Для пользователей с нарушениями слуха

Видеоматериалы:

- Включите субтитры для всех видеозаписей, чтобы обеспечить доступность информации.

- Используйте жестовый язык в видео для объяснения ключевых моментов.

Текстовые инструкции:

- Предоставьте письменные инструкции, которые можно использовать как дополнение к видео.

- Убедитесь, что текст доступен в разных форматах (PDF, HTML).

3. Адаптация обучающих форматов

3.1. Многообразие форматов

- Комбинирование форматов: Используйте текст, видео, аудио и графику в одном обучающем материале для обеспечения максимальной доступности.

- Интерактивные элементы: Создавайте интерактивные обучающие модули, которые можно адаптировать под разные потребности.

3.2. Простота навигации

- Четкая структура: Обеспечьте логичную структуру материалов, чтобы пользователи могли легко находить нужную информацию.

- Обозначение разделов: Используйте заголовки и подзаголовки для упрощения поиска информации.

4. Тестирование и обратная связь

4.1. Привлечение целевой аудитории

- Пилотное тестирование: Проведите тестирование материалов с участием людей с нарушениями зрения и слуха.

- Сбор отзывов: Опросите участников о том, насколько материалы были понятны и доступны.

4.2. Корректировка материалов

- Внесение изменений: На основе полученной обратной связи обновите материалы для улучшения доступности.

5. Поддержка и ресурсы

5.1. Дополнительные ресурсы

- Горячая линия поддержки: Организуйте канал для вопросов и получения помощи по обучающим материалам.

- Часто задаваемые вопросы: Создайте раздел с ответами на распространенные вопросы, доступный в текстовом формате.

# ГЛАВА 6. ПРАКТИКА – КРИТЕРИЙ ИСТИНЫ. ПРАКТИЧЕСКОЕ ПРИМЕНЕНИЕ МЕТОДИК ТЕСТИРОВАНИЯ ЭЛЕМЕНТОВ ДОСТУПНОЙ СРЕДЫ В ПРИЛОЖЕНИЯХ

Эта глава нашей книги посвящена практическому применению методик тестирования элементов доступной среды в приложениях. Здесь мы сосредоточимся на том, как теоретические знания о цифровой доступности могут быть эффективно реализованы на практике. Мы рассмотрим различные практические подходы к тестированию, которые позволяют выявить и устранить барьеры, мешающие пользователям с ограниченными возможностями.

В этой главе опираясь на рассмотренные методики и инструменты, используемые для оценки доступности приложений, мы рассмотрим практические примеры успешного применения этих методик в реальных проектах.

Кроме того, мы уделим внимание критериям оценки доступности и тому, как они могут служить основой для принятия решений в процессе разработки. Важно понимать, что доступность — это не одноразовое мероприятие, а постоянный процесс, требующий регулярного анализа и корректировки на всех этапах жизненного цикла продукта.

В заключение главы мы подведем итоги и выделим основные выводы, которые помогут разработчикам и дизайнерам создать более инклюзивные приложения. Мы уверены, что сочетание теории и практики позволит нам не только повысить уровень цифровой доступности, но и внести значимый вклад в создание более справедливого и равноправного общества.

Рассмотрим пример тестирования веб-приложения для заказа продуктов с акцентом на доступность для людей с нарушениями зрения и слуха.

1. Подготовка к тестированию

1.1. Определение целей

- Проверить, насколько приложение соответствует стандартам доступности (WCAG 2.1).

- Убедиться, что пользователи с нарушениями зрения и слуха могут легко заказывать продукты.

1.2. Сбор инструментов

- Для пользователей с нарушениями зрения: JAWS, NVDA, Axe.

- Для пользователей с нарушениями слуха: WAVE, VoiceOver для проверки субтитров и транскрипций.

2. Тестирование доступности для людей с нарушениями зрения

2.1. Проверки правильной семантической разметки HTML.

2.2. Проверка правильной логической структуры и заголовков HTML

2.3. Использование экранных читалок

- Тестирование навигации:

 - Проверьте, может ли пользователь перемещаться по основным разделам (категории продуктов, корзина, оформление заказа) с помощью клавиатуры.

- Проверка ARIA-меток:

 - Убедитесь, что кнопки «Добавить в корзину», «Оформить заказ» имеют правильные ARIA-метки, такие как aria-label.

2.4. Проверка изображений

- Все изображения продуктов должны иметь атрибуты alt, описывающие продукт (например, «Яблоко красное, 1 кг»).

2.5. Проверка форм

- Убедитесь, что все поля формы (например, имя, адрес доставки) имеют метки и подсказки, доступные для экранных читалок.

- Проверьте наличие сообщений об ошибках (например, «Поле обязательно для заполнения») и их доступность.

2.6. Проверка контраста

- Используйте Color Contrast Checker для проверки контрастности текста и фона на страницах приложения.

3. Тестирование доступности для людей с нарушениями слуха

3.1. Проверка мультимедиа

- Если есть видеообзоры продуктов, убедитесь, что они имеют точные субтитры.

3.2. Проверка уведомлений

- Убедитесь, что важные уведомления (например, ошибки при оформлении заказа) отображаются как визуально (всплывающее сообщение), так и через звуковые сигналы. Предоставьте текстовые сообщения для пользователей с нарушениями слуха.

4. Функциональная проверка

4.1. Тестирование пользовательского интерфейса

- Убедитесь, что все элементы управления (кнопки, выпадающие списки) доступны через клавиатуру и имеют визуальные индикаторы фокуса.

4.2. Проверка на мобильных устройствах

- Проверьте доступность интерфейса на мобильных устройствах с использованием экранных читалок.

5. Заключение

5.1. Составление отчета

- Задокументируйте все найденные проблемы:
  - Неправильные или отсутствующие ARIA-метки.
  - Недостаточный контраст текста.
  - Отсутствие альтернативного текста для изображений.

- Проблемы с навигацией через клавиатуру.

5.2. Повторное тестирование

- После внесения изменений проведите повторное тестирование, чтобы убедиться в исправлении всех проблем.

Итог

Тестирование доступности веб-приложения для заказа продуктов требует внимательного подхода к деталям, чтобы обеспечить удобство использования для всех пользователей, независимо от их ограничений. Регулярное тестирование и обновление приложения помогут поддерживать его соответствие стандартам доступности.

# ЗАКЛЮЧЕНИЕ

В заключении книги подведем итоги нашего глубокого исследования важности доступности в цифровом мире. Мы рассмотрели, как доступная среда становится неотъемлемой частью современных приложений, обеспечивая равные возможности для всех пользователей, включая людей с ограниченными возможностями.

На протяжении книги мы проанализировали ключевые стандарты и требования к цифровой доступности, такие как WCAG и ADAAG, а также представили рекомендации по их выполнению. Мы также обсудили различные инструменты тестирования, которые могут помочь разработчикам и тестировщикам в обеспечении доступности веб-контента и приложений.

Методика тестирования, представленная в книге, охватывает все этапы разработки — от предварительного тестирования до подготовки релиза. Автор подчеркнул важность интеграции аспектов доступности на ранних стадиях разработки и предложили практические подходы к проверке и улучшению доступности программных продуктов.

Практическое применение методик тестирования, описанных в последних главах, демонстрирует, что доступная среда не только улучшает пользовательский опыт, но и способствует созданию более инклюзивного общества. Реализация доступных решений — это не просто выполнение стандартов; это шаг к социальной ответственности и уважению к каждому пользователю.

Автор надеется, что эта книга станет полезным ресурсом для разработчиков, тестировщиков и всех заинтересованных в создании доступных приложений. Доступная среда — это не только требование времени, но и возможность для инноваций и роста в области технологий.

Создавая приложения с учетом потребностей всех пользователей, мы вместе можем сделать мир более открытым и доступным для всех.

## СПИСОК ИСПОЛЬЗОВАННОЙ ЛИТЕРАТУРЫ

1. Баранов, И. А. Доступная среда: теория и практика / И. А. Баранов. — Москва: Издательство «Наука», 2020. — 256 с.

2. Васильева, Т. П. Методика тестирования программного обеспечения / Т. П. Васильева. — Санкт-Петербург: Издательство «Питер», 2019. — 312 с.

3. Громова, Е. В., Сидоров, А. Н. Инклюзивные технологии в образовании: от теории к практике / Е. В. Громова, А. Н. Сидоров. — Екатеринбург: УралГТУ, 2021. — 384 с.

4. Кузнецов, С. В. Доступность веб-приложений для людей с ограниченными возможностями / С. В. Кузнецов. — Казань: Издательство «Казанский университет», 2018. — 220 с.

5. Лебедев, А. И., Петрова, М. А. Основы доступной среды: методические рекомендации / А. И. Лебедев, М. А. Петрова. — Новосибирск: Сибирское университетское издательство, 2022. — 150 с.

6. Михайлова, О. В., Смирнов, И. Ю. Тестирование программного обеспечения с учетом доступности / О. В. Михайлова, И. Ю. Смирнов. — Ростов-на-Дону: Издательство «Феникс», 2021. — 290 с.

7. Петров, А. С., Коваленко, Н. В. Доступная среда в информационных технологиях / А. С. Петров, Н. В. Коваленко. — Москва: Издательство «Инфра-М», 2020. — 350 с.

8. Соловьев, Д. А., Федорова, Л. И. Инклюзивные технологии: от теории к практике / Д. А. Соловьев, Л. И. Федорова. — Минск: Издательство «Белорусский университет», 2019. — 400 с.

9. Тихомиров, Р. П., Яковлева, Е. В. Практика тестирования доступности веб-приложений / Р. П. Тихомиров, Е. В. Яковлева. — Владивосток: Дальневосточное издательство, 2021. — 275 с.

10. Шевченко, Н. Г., Иванов, К. А. Создание доступной среды для пользователей с ограниченными возможностями / Н. Г. Шевченко, К. А. Иванов. — Челябинск: Южноуральский государственный университет, 2022. — 310 с.

11. [Axe Accessibility Testing Tool](https://www.deque.com/axe/)

12. [WAVE Evaluation Tool](https://wave.webaim.org/)