МИНОБРНАУКИ РОССИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ

ВЫСШЕГО ОБРАЗОВАНИЯ

«НИЖЕГОРОДСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ

им. Р.Е. АЛЕКСЕЕВА»

(НГТУ)

|  |  |  |
| --- | --- | --- |
| Институт | Институт радиоэлектроники и информационных технологий | |
| Направление подготовки | 09.03.02 Информационные системы и технологии | |
| Направленность (профиль) образовательной программы | | Информационные технологии в дизайне |
| Кафедра | Графические информационные системы | |

**ОТЧЕТ**

**по прохождению Ознакомительной практики**

|  |  |
| --- | --- |
| Выполнил |  |
| Студент гр. 23-ИСТ-4-1 | Зинин Р. Ю. |
| Руководитель практики от кафедры Филинских А.Д. | |
| Степень, звание | к.т.н., доцент |
|  |  |
|  |  |
| Отчет защищен с оценкой | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |
| Дата защиты | «19» июля 2024г. |

Нижний Новгород

2024 год

Оглавление

[ИНДИВИДУАЛЬНОЕ ЗАДАНИЕ 3](#_Toc44685738)

[Введение. 8](#_Toc44685739)

[Основная часть отчета в соответствии с заданием на практику 6](#_Toc44685740)

[Заключение 26](#_Toc44685741)

[Список литературы 27](#_Toc44685743)

[Приложение 28](#_Toc44685744)

МИНОБРНАУКИ РОССИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ

ВЫСШЕГО ОБРАЗОВАНИЯ

«НИЖЕГОРОДСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ

им. Р.Е. АЛЕКСЕЕВА»

(НГТУ)

Кафедра Графические информационные системы

|  |  |
| --- | --- |
|  | Утверждаю: |
|  | Заведующий кафедрой |
|  | А.Д. Филинских |
|  | «06» июля 2024 г. |

# ИНДИВИДУАЛЬНОЕ ЗАДАНИЕ

на Учебную практику (Ознакомительная практика)

Студента гр. 23-ИСТ-4-1 Зинина Романа Юрьевича.

Направление подготовки/специальность: 09.03.02 Информационные системы и технологии

Профиль подготовки: Информационные технологии в дизайне

Место прохождения практики НГТУ им. Р.Е. Алексеева, кафедра «Графические информационные системы»

Время прохождения практики

Дата начала практики «06» июля 2024 г.

Дата окончания практики «19» июля 2024 г.

**Тема индивидуального задания:**

**Создание сайта с использованием всплывающей анимации.**

Содержание практики

*Во время прохождения практики студент обязан:*

**Ознакомиться:** Ознакомиться с концепциями программирования JavaScript, HTML

**Изучить:** Применения языков программирования JavaScript и HTML для создания сайтов.

**Выполнить следующие виды работ по приобретению практических навыков:**

Создание сайта с использованием всплывающей анимации при нажатии на кнопку.

Должность на практике практикант

Планируемые результаты обучения при прохождении практики

| **Планируемые результаты освоения образовательной программы** | **Планируемые результаты обучения при прохождении практики** | | | |
| --- | --- | --- | --- | --- |
| **Код компетенции** | **Знать** | **Уметь** | **Владеть** | |
| ОПК-1 Способен применять естественнонаучные и общеинженерные знания, методы математического анализа и моделирования, теоретического и экспериментального исследования в профессиональной деятельности | * основы математики, физики, вычислительной техники и программирования; | * решать стандартные профессиональные задачи с применением естественнонаучных и общеинженерных знаний, методов математического анализа и моделирования; | * навыками теоретического и экспериментального исследования объектов профессиональной деятельности; |
| ОПК-3 Способен решать стандартные задачи профессиональной деятельности на основе информационной и библиографической культуры с применением информационно-коммуникационных технологий и с учетом основных требований информационной безопасности; | * принципы, методы и средства решения стандартных задач профессиональной деятельности на основе информационной и библиографической культуры с применением информационно-коммуникационных технологий и с учетом основных требований информационной безопасности | - решать стандартные задачи профессиональной деятельности на основе информационной и библиографической культуры с применением информационно-коммуникационных технологий и с учетом основных требований информационной безопасности | * навыками подготовки обзоров, аннотаций, составления рефератов, научных докладов, публикаций и библиографии по научно-исследовательской работе с учетом требований информационной безопасности | |
| ОПК-6 Способен разрабатывать алгоритмы и программы, пригодные для практического применения в области информационных систем и технологий; | |  | | --- | | * основные языки программирования и работы с базами данных, операционные системы и оболочки, современные программные среды разработки информационных систем и технологий | | * применять языки программирования и работы с базами данных, современные программные среды разработки информационных систем и технологий для автоматизации бизнес-процессов, решения прикладных задач различных классов, ведения баз данных и информационных хранилищ | * навыками программирования, отладки и тестирования прототипов программно-технических комплексов задач | |

Руководитель практики от кафедры

Зав. каф. ГИС к.т.н, доцент Филинских А.Д,

(ученое звание и степень) (подпись) Ф.И.О.

Задание на практику получил:

Студент \_\_\_\_\_\_\_\_\_\_\_\_\_ Зинин Роман Юрьевич

(подпись) (ФИО)

«06» июля 202 г.

**РАБОЧИЙ ГРАФИК (ПЛАН)**

**ПРОВЕДЕНИЯ Учебной практики**

Студента гр. 23-ИСТ-4-1 Ф.И.О. Зинин Роман Юрьевич

| №№ п/п | Разделы (этапы) практики  (заполняется в соответствии с заданием) | Сроки выполнения  с «5» июля 2024 по «19» июля 2024 | Отметка о выполнении (подпись руководителя практики) |
| --- | --- | --- | --- |
| 1 | **Подготовительный (организационный) этап** | 06 июля 2024г  07 июля 2024г. |  |
| 1.1 | Определение базы прохождения практики | 06 июля 2024 |  |
| 1.2 | Организационное собрание для разъяснения целей, задач, содержания и порядка прохождения практики | 07 июля 2024 |  |
| 1.3 | Получение индивидуального задания |  |  |
| 2 | **Выполнение индивидуального задания:** | 08 июля 2024г  17 июля 2024г |  |
| 2.1 | Разработка идеи сайта. | 7 июля 2024 г |  |
| 2.2 | Заполнение тз | 9 июля 2024 г |  |
| 2.3 | Разработка дизайна сайта в фигма | 11 июля 2024 г |  |
| 2.4 | Разработка сайта в соответствии с дизайном | 13 июля 2024 г |  |
| 2.5 | Создание анимации на сайте | 15 июля 2024 г |  |
| 2.6 | Создание JavaScript кода для организации логики сайта и анимации | 17 июля 2024 г |  |
| 2.6 | Доработка JavaScript кода | 18 июля 2024 г |  |
| 2.6 |  |  |  |
| 2.7 |  |  |  |
| 2.8 |  |  |  |
| 2.9 |  |  |  |
| 3 | Заключительный этап | 18 июля 202\_\_г  19 июля 202\_\_г |  |
| 3.1 | *Написание и оформление отчета по практике* | 18 июля 202\_\_г |  |
| 3.2 | *Защита отчета по практике* | 19 июля 202\_\_г |  |

Руководитель практики от кафедры

доцент, к.т.н. \_\_\_\_\_\_\_\_\_\_\_\_\_ \_Филинских А.Д,\_\_\_\_\_\_\_\_\_

*(ученые звание и степень) (подпись)* Ф.И.О.

# Введение.

**Цель:**

Цель практики заключается в обучении созданию анимаций всплывающих элементов на веб-сайте с использованием HTML и JavaScript. На современном веб-сайте анимация играет важную роль в улучшении пользовательского опыта, делая взаимодействие с сайтом более плавным и интуитивно понятным. HTML предоставляет основу для структуры веб-страницы, в то время как JavaScript добавляет динамическое поведение и позволяет контролировать анимации. В ходе практики я изучил, как правильно использовать эти технологии для создания анимаций, которые будут активироваться при определенных действиях пользователя, таких как нажатие кнопки или загрузка страницы. Это включает в себя понимание принципов работы с DOM (Document Object Model), применение CSS для стилизации анимаций и использование JavaScript для управления состоянием элементов и их анимацией. В результате, я научился создавать более привлекательные и интерактивные веб-сайты, которые смогут эффективно удерживать внимание пользователей и улучшать общий пользовательский опыт.

**Задачи:**

1. Освоить основные принципы программирования на языке JavaScript.
2. Овладеть методами научно-технического творчества для решения профессиональных задач.
3. Научиться применять компьютерные методы для сбора, хранения и обработки информации в профессиональной деятельности.
4. Изучить различные подходы и технологии для создания и организации веб-страниц.
5. Разработать веб-страницу, используя приобретенные знания и навыки.

**Список компетенций:**

*Системные:*

* Применение знаний на практике;
* Улучшение исследовательских навыков в программировании;
* Создание и управление собственным проектом;
* Развитие способности генерировать и реализовывать новые идеи.
* Эффективное использование инновационных технологий в профессиональной деятельности.

*Инструментальные:*

* Основные знания в различных областях программирования;
* Навыки эффективного управления информацией (способность искать и анализировать данные из различных источников);
* Решение сложных проблем и принятие самостоятельных решений.
* Дополнительный пункт:
* Применение аналитического мышления для оптимизации процессов и улучшения результативности.

# Основная часть отчета в соответствии с заданием на практику

Учебная практика по разработке и сопровождению программного обеспечения способствовала получению навыков работы с веб-страницами, применение каскадных таблиц стилей и программирование на языке JavaScript.

*Задача проекта*: создание сайта с применением всплывающей анимации элементов сайта, для большей интерактивности. Сайт должен быть удобен в использовании, т.к. предполагаемая аудитория – дети.

*Сайт должен обеспечивать*:

* Простую навигацию по сайту.
* Простой доступ к легко запоминающейся информации.
* Возможность проверить только что полученные знания по средствам прохождения короткой интерактивной викторины.
* Возможность получить результат викторины

*Веб-страницы должны содержать:*

* *Большие вызывающие надписи, для простого взаимодействия.*
* *Форму с информацией*
* *Кнопки для навигации*
* *Форму с викториной*
* *Форму с результатами викторины*

*Этапы реализации:*

Этап 1: Разработка идеи самого проекта и создание дизайн проекта в программе figma.



Рисунок 1

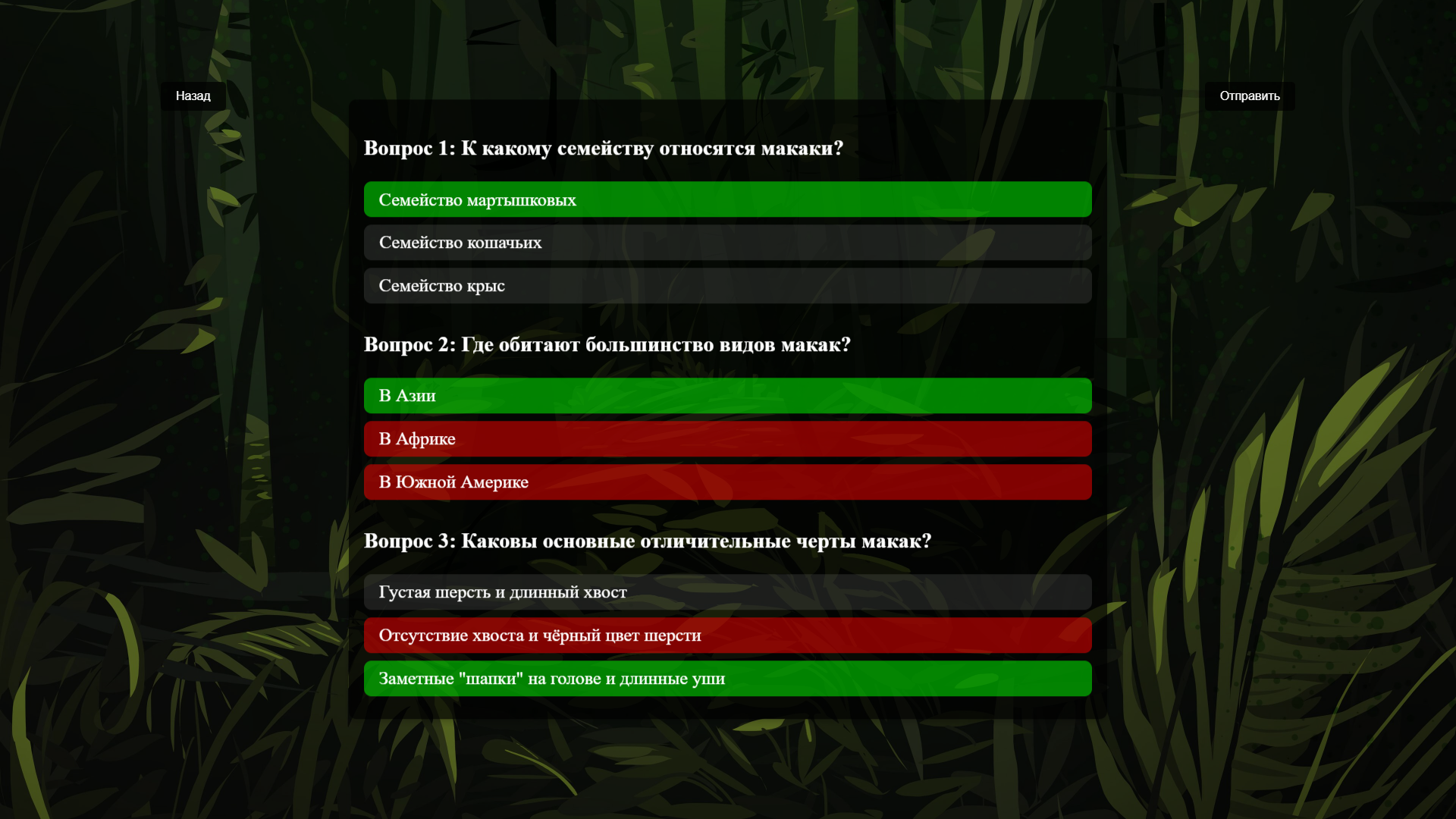


Рисунок 2

Этап 2: Реализация проекта с помощью HTML, CSS и JS.

*HTML* – это язык разметки страницы (говорит браузеру, какие элементы должны находиться на странице и в каком месте).

HTML и CSS играют ключевые роли в создании веб-страниц. HTML (HyperText Markup Language) определяет структуру содержимого: он использует теги для разметки различных элементов и их функций на веб-странице. В свою очередь, CSS (Cascading Style Sheets) отвечает за внешний вид и стиль страницы. Если HTML можно сравнить с каркасом или основной структурой, то CSS можно назвать краской, которая добавляет цвет, текстуры и общий визуальный стиль.

CSS действительно мощный инструмент, позволяющий задавать цвета, шрифты, отступы и расположение элементов на странице. Он также обеспечивает адаптивность веб-сайтов к разным размерам экранов и устройствам.

Термин "каскадные таблицы стилей" отражает основной принцип CSS: возможность наследования и переопределения стилей на разных уровнях. Каждый веб-браузер имеет свои стандартные таблицы стилей, но разработчики могут создавать и подключать собственные таблицы стилей, что позволяет контролировать внешний вид страницы независимо от её содержимого и используемого браузера.

Кроме того, основной задачей проекта в первую очередь являются анимации. CSS помогает реализовывать анимации и интерактивные эффекты на веб-страницах. Это достигается благодаря возможностям CSS для управления свойствами элементов, такими как положение, размер, цвет и прозрачность, что делает его важным инструментом в современной веб-разработке.

JavaScript добавляет сайту функциональность и интерактивность, что делает его более динамичным и удобным для пользователей. Этот язык программирования, разработанный в 1995 году, является ключевым инструментом для создания сложных функций и обработки событий на веб-страницах.

JavaScript позволяет реализовывать различные возможности, такие как проверка форм, анимации, динамическое обновление содержимого страницы без необходимости перезагрузки, валидацию данных, манипуляции с DOM (Document Object Model) и многое другое. Это делает его неотъемлемой частью современной веб-разработки.

JavaScript также широко используется для создания веб-приложений, включая сложные интерфейсы, игры, эффекты визуализации данных и многопользовательские системы. Благодаря тому, что JavaScript поддерживается всеми современными браузерами, разработчики могут быть уверены в том, что их функциональность будет работать на всех платформах и устройствах.

В заключение, JavaScript существенно расширяет возможности веб-страниц и делает интернет-пространство более интерактивным и увлекательным для пользователей.

**Рассмотрим код HTML.**

*Страница 1.*

1. **<div class="backgroundGif"></div>**

* Этот div не содержит контента напрямую и используется для отображения фоновой графики формата **Gif** средствами CSS и JavaScript.

1. **<div class="topText" onclick="toggleInfo()">Читать О макаках</div>**

* Этот div является кликабельным элементом ("Читать О макаках"), который при клике вызывает функцию toggleInfo(). Эта функция показывает или скрывает дополнительную информацию о макаках.

1. **<div class="mon1" id="infoBlock"></div>**

* Этот div (infoBlock) содержит основной контент о макаках. Внутри него располагаются элементы, такие как изображения и текстовые блоки, обрабатываемые другими функциями JavaScript.

1. **<div class="monPaj" onclick="swingImage()" onmouseover="toggleImageSize()" onmouseout="resetImageSize()"></div>**

* Этот div служит для отображения изображения макаки, которое можно анимировать при клике (swingImage()) и изменять размер при наведении (toggleImageSize() и resetImageSize()).

1. **<div class="textMAKAK"></div>**

* Этот div содержит текстовую информацию о макаках. Внутри него есть абзацы с описанием и возможностью раскрыть дополнительный текст при клике на ссылку ("Подробнее...").

1. **<div class="additionalText" id="additionalText"></div>**

* Этот div скрыт и может быть раскрыт при клике на ссылку "Подробнее...". Он содержит дополнительное описание о макаках.

1. **<div class="mon2" id="quizBlock" onclick="expandQuizBlock()"></div>**

* Этот div представляет собой элемент, при клике на который запускается функция expandQuizBlock(). Он используется для раскрытия блока с викториной.

*Страница 2.*

1. <div class="quizOpros" id="quizOpros"></div>

* Этот div содержит блок с вопросами и ответами в викторине. Он имеет id "quizOpros", что может использоваться для стилизации и для управления содержимым с помощью JavaScript.

2. <div class="quizQuestion"></div>

* Этот div представляет собой блок с отдельным вопросом в викторине.

3. <h3></h3>

* Заголовок вопроса. Каждый вопрос обрамлен в тег <h3></h3>, чтобы подчеркнуть его значимость и разделение от других вопросов.

4. <div class="quizAnswers"></div>

* Этот div содержит варианты ответов на конкретный вопрос в виде радиокнопок.

5. <input type="radio"></input>

* Радиокнопки, которые позволяют выбрать один из вариантов ответа на вопрос. У радиокнопок, которые содержат правильный ответ, указан атрибут data-correct.

6. <label for="..."></label>

* Метки для радиокнопок, предоставляющие текстовое описание вариантов ответа. Атрибут for связывает метку с соответствующим элементом ввода.

7. <div class="buttonContainer" id="buttonContainer"></div>

* Этот div содержит кнопки управления викториной, такие как кнопка "Назад" (backButton) и кнопка "Отправить" (submitButton). На кнопке "Отправить" изначально установлен класс submitButtonHidden, что скрывает её до момента подведения итогов.

8. <div class="results" id="results"></div>

* Этот div предназначен для отображения результатов викторины после её завершения. Включает заголовок "Результаты теста" и блоки для отображения баллов и статистики первой попытки.

**Рассмотрим код CSS.**

*Страница 1.*

Продолжение описания основных селекторов и их функционала:

1. .backgroundGif

* Этот класс применяется к абсолютно позиционированному элементу, который занимает всю область родительского элемента (body). Он отображает фоновую графику изображением (background.gif), которая занимает всю доступную площадь и масштабируется под размеры экрана.

2. .topText

* Этот класс стилизует текстовый элемент, расположенный над фоновой графикой. Текст центрируется по горизонтали, имеет белый цвет, большой размер шрифта (8em), устанавливает курсор в виде указателя при наведении (cursor: pointer) и запрещает выделение текста пользователем (user-select: none). При наведении на элемент применяется анимация pulseAnimation.

3. .mon1

* Этот класс применяется к блоку с информацией о макаках. Он имеет заданные размеры (500px высотой и 80% ширины), полупрозрачный чёрный фон (rgba(0, 0, 0, 0.5)), закруглённые углы (20px), абсолютное позиционирование и переходное свойство для анимированного перехода по горизонтали (transition: left 0.5s ease). При добавлении класса .show, элемент перемещается влево на 10% ширины страницы.

4. .mon2

* Этот класс стилизует блок для кнопки "Начать викторину". Он имеет фон аналогичного полупрозрачного чёрного цвета, абсолютное позиционирование, расположение по центру внизу страницы, белый текст, курсор в виде указателя и анимацию pulseAnimation при наведении на текст внутри блока. При добавлении класса .show для .mon1, блок .mon2 поднимается вверх на 10vh.

5. .monPaj

* Этот класс применяется к элементу, представляющему изображение макаки. Он имеет заданные размеры (max-width: 500px), фоновое изображение (1.png), которое масштабируется и позиционируется по центру блока, и анимацию transform для уменьшения масштаба при наведении курсора.

Анимации:

1. @keyframes pulseAnimation

* Эта анимация создаёт эффект пульсации элемента: начинается с полной видимости (opacity: 1), затем на 50% анимации элемент становится менее прозрачным (opacity: 0.5), и возвращается к полной видимости к концу анимации.

2. @keyframes swingAnimation

* Эта анимация предназначена для элемента .monPaj, чтобы добавить эффект качания изображения: элемент начинает с нулевого поворота и уменьшенного масштаба, затем поворачивается в разные стороны на разных этапах анимации.

3. @keyframes textAnimationShow и @keyframes textAnimationHide

* Эти анимации используются для плавного появления и исчезновения дополнительного текста о макаках (additionalText). Они изменяют opacity и размер шрифта текста для достижения эффекта плавности.

*Страница 2.*

Продолжение описания основных селекторов и их функционала:

1. quizOpros

* Этот класс применяется к блоку, содержащему вопросы и ответы в викторине. Он абсолютно позиционирован по центру экрана с помощью top: 50%; left: 50%; transform: translate(-50%, -50%);. Имеет полупрозрачный черный фон (rgba(0, 0, 0, 0.7)), округленные углы (border-radius: 10px), тень (box-shadow), белый текст и установленный размер шрифта (1.5em). Элемент по умолчанию скрыт (opacity: 0).

2. .quizAnswers

* Этот класс задает стили для контейнера с ответами в викторине. Он использует flex-direction: column для вертикального расположения ответов.

3. .quizAnswers label

* Этот селектор стилизует ответы в викторине. Они имеют полупрозрачный белый фон (rgba(255, 255, 255, 0.1)), закругленные углы (border-radius: 10px), отступы и курсор в виде указателя при наведении.

4. .quizAnswers input[type="radio"]

* Этот селектор скрывает стандартные радиокнопки в ответах в викторине.

5. .quizAnswers input[type="radio"]:checked + label

* Этот селектор стилизует выбранный пользователем правильный ответ в викторине. Он изменяет фоновый цвет на зеленый (rgba(0, 255, 0, 0.5)).

6. .quizAnswers input[type="radio"]:checked:not([data-correct]) + label

* Этот селектор стилизует выбранный пользователем неправильный ответ в викторине. Он изменяет фоновый цвет на красный (rgba(255, 0, 0, 0.5)).

7. .quizQuestion

* Этот класс применяется к блоку с вопросом в викторине. Он устанавливает начальную непрозрачность (opacity: 0) и используется в анимации для появления блока.

Анимации:

1. .slideInLeft и .slideInRight

* Эти классы используются для анимации появления блоков вопросов слева и справа соответственно. Они применяют анимацию slideInLeft и slideInRight на 1 секунду.

2. @keyframes slideInLeft и @keyframes slideInRight

* Эти анимации задают движение элементов при их появлении: они начинают с непрозрачности и сдвига влево или вправо (зависит от анимации) за 1 секунду.

3. .fallIn

* Этот класс применяется для анимации появления элемента снизу вверх (top от -100% до 50%) с небольшими плавными колебаниями перед окончательным фиксированием. Применяется анимация fallIn на 1 секунду.

4. @keyframes fallIn

* Эта анимация управляет появлением элемента снизу вверх с плавным изменением его позиции и непрозрачности.

5. .buttonsVisible и .submitButtonVisible

* Эти классы используются для показа кнопок викторины и кнопки отправки ответов. Они устанавливают opacity: 1 и display: block соответственно после выполнения условий или анимаций.

6. .submitButtonHidden

* Этот класс используется для скрытия кнопки отправки ответов. Он устанавливает display: none.

7. .results

* Этот класс применяется к блоку результатов теста. Он абсолютно позиционирован по центру экрана, имеет полупрозрачный черный фон, округленные углы, белый текст и установленный размер шрифта. Элемент по умолчанию скрыт (display: none и opacity: 0), но анимацией fadeIn он появляется на экране.

8. @keyframes fadeIn

* Эта анимация управляет появлением блока результатов: элемент начинает с непрозрачности 0 и постепенно становится полностью видимым за 1 секунду.

**Рассмотрим код JavaScript.**

*Страница 1.*

1. Переменная originalTransform

* Эта переменная используется для хранения исходного значения свойства transform, чтобы можно было вернуть изображение к его первоначальному размеру после изменения масштаба.

2. Функция toggleImageSize()

* Данная функция уменьшает размер изображения (элемента с классом monPaj). Сначала она сохраняет текущее значение transform в переменную originalTransform, затем устанавливает масштаб изображения до 0.8 с помощью свойства transform.

// Функция для уменьшения размера изображения

function toggleImageSize() {

var monPaj = document.querySelector('.monPaj'); // Выбираем элемент с классом 'monPaj' и сохраняем его в переменную monPaj

originalTransform = monPaj.style.transform; // Сохраняем текущее значение transform

monPaj.style.transform = 'scale(0.8)'; // Устанавливаем масштаб 0.8

}

3. Функция resetImageSize()

* Эта функция восстанавливает исходный размер изображения, возвращая ему значение transform, сохраненное в originalTransform.

// Функция для восстановления исходного размера изображения

function resetImageSize() {

var monPaj = document.querySelector('.monPaj'); // Выбираем элемент с классом 'monPaj' и сохраняем его в переменную monPaj

monPaj.style.transform = originalTransform; // Восстанавливаем исходное значение transform

}

4. Функция swingImage()

* Эта функция добавляет анимацию "качели" изображению (monPaj). Она устанавливает анимацию swingAnimation с продолжительностью 0.5 секунды и затем очищает анимацию через 0.5 секунды с помощью setTimeout.

// Функция для анимации качели изображения

function swingImage() {

var monPaj = document.querySelector('.monPaj'); // Выбираем элемент с классом 'monPaj' и сохраняем его в переменную monPaj

monPaj.style.animation = 'swingAnimation 0.5s ease'; // Запускаем анимацию swingAnimation

setTimeout(function() {

monPaj.style.animation = ''; // Очищаем анимацию после 0.5 секунды

}, 500);

}

5. Функция toggleAdditionalText()

* Эта функция переключает дополнительный текст, связанный с элементом с id additionalText. Если текст видим и отображается, функция скрывает его с анимацией textAnimationHide и устанавливает его display на none. Если текст скрыт, функция показывает его с анимацией textAnimationShow и устанавливает его display на block.

// Функция для переключения дополнительного текста

function toggleAdditionalText() {

var additionalText = document.getElementById('additionalText'); // Получаем элемент по его id 'additionalText' и сохраняем его в переменную additionalText

if (additionalText.style.display === 'block') {

// Если текст отображается, скрываем его с анимацией

additionalText.style.animation = 'textAnimationHide 0.5s forwards'; // Применяем анимацию textAnimationHide длительностью 0.5 секунды с заполнением состояния

setTimeout(function() {

additionalText.style.display = 'none'; // Скрываем элемент после завершения анимации через 0.5 секунды

}, 500);

} else {

// Если текст скрыт, показываем его с анимацией

additionalText.style.display = 'block'; // Отображаем элемент

additionalText.style.animation = 'textAnimationShow 0.5s forwards'; // Применяем анимацию textAnimationShow длительностью 0.5 секунды с заполнением состояния

}

}

6. Функция toggleInfo()

* Эта функция переключает информационный блок (infoBlock). Если блок уже показан (infoBlock содержит класс show), функция убирает этот класс и сдвигает блок викторины (quizBlock) вниз до 50vh. Если блок скрыт, функция добавляет класс show к infoBlock, отображая его, и сдвигает quizBlock вниз до 10vh.

// Функция для переключения информационного блока

function toggleInfo() {

var infoBlock = document.getElementById('infoBlock'); // Получаем элемент с id 'infoBlock' и сохраняем его в переменную infoBlock

var quizBlock = document.getElementById('quizBlock'); // Получаем элемент с id 'quizBlock' и сохраняем его в переменную quizBlock

if (infoBlock.classList.contains('show')) {

// Если информационный блок показан, скрываем его

infoBlock.classList.remove('show'); // Удаляем класс 'show' у элемента infoBlock, чтобы скрыть блок

quizBlock.style.bottom = '50vh'; // Устанавливаем свойство bottom для quizBlock на 50% высоты окна, чтобы корректно отображать элемент под информационным блоком

} else {

// Если информационный блок скрыт, показываем его

infoBlock.classList.add('show'); // Добавляем класс 'show' элементу infoBlock, чтобы показать блок

quizBlock.style.bottom = '10vh'; // Устанавливаем свойство bottom для quizBlock на 10% высоты окна, чтобы поднять его над информационным блоком

}

}

7. Функция expandQuizBlock()

* Эта функция выполняет перенаправление пользователя на другую страницу с тестом. Она использует window.location.href для загрузки новой страницы. В данном случае, она перенаправляет на страницу 'test1.htm'.

function expandQuizBlock() {

// Перенаправление на другую страницу

window.location.href = 'test1.htm';

}

*Взаимодействие с HTML и CSS.*

* Функции toggleImageSize(), resetImageSize(), swingImage() и toggleAdditionalText() взаимодействуют с элементами, определенными в HTML, такими как .monPaj и #additionalText, изменяя их стили через свойства JavaScript (style.transform, style.animation, style.display).
* Функция toggleInfo() изменяет классы и стили элемента infoBlock в зависимости от его текущего состояния (classList.add() и classList.remove()), а также управляет положением quizBlock.
* Функция expandQuizBlock() работает с URL-адресом (window.location.href), перенаправляя пользователя на новую страницу теста.

Этот JavaScript код обеспечивает интерактивность элементов веб-страницы, включая анимацию, изменение размера изображения и переключение текстовых блоков, что делает пользовательский опыт более динамичным и привлекательным.

*Страница 2.*

1. Функция calculateScore()

* Эта функция перебирает все вопросы в викторине, проверяет, какие ответы выбраны, отмечает правильные и неправильные ответы цветом, подсчитывает количество правильных ответов и ошибок при первой попытке. Затем она выводит результаты в консоль и обновляет страницу, показывая количество правильных ответов и ошибок.

// Функция для подсчета правильных и неправильных ответов

function calculateScore() {

var totalQuestions = quizQuestions.length; // Общее количество вопросов

var correctAnswers = 0; // Счетчик правильных ответов

// Перебираем все вопросы

quizQuestions.forEach(function(question) {

var radioButtons = question.querySelectorAll('input[type="radio"]');

var isCorrect = false;

var isFirstAttempt = true;

// Перебираем радио-кнопки в текущем вопросе

radioButtons.forEach(function(radio) {

if (radio.checked) {

var label = radio.nextElementSibling;

// Проверяем, правильный ли ответ выбран

if (radio.hasAttribute('data-correct')) {

label.style.backgroundColor = 'rgba(0, 255, 0, 0.5)'; // Отмечаем правильный ответ зеленым

correctAnswers++;

isCorrect = true;

} else {

label.style.backgroundColor = 'rgba(255, 0, 0, 0.5)'; // Отмечаем неправильный ответ красным

// Если это первая попытка и ответ неправильный, увеличиваем счетчик ошибок

if (isFirstAttempt) {

firstAttemptWrong++;

}

}

isFirstAttempt = false; // После первого выбора устанавливаем false

}

});

// Если выбран правильный ответ, отключаем все радио-кнопки в текущем вопросе

if (isCorrect) {

radioButtons.forEach(function(radio) {

radio.disabled = true;

});

}

});

// Выводим результаты в консоль

console.log('Количество правильных ответов: ' + correctAnswers + ' из ' + totalQuestions);

console.log('Ошибки при первых попытках: ' + firstAttemptWrong);

// Показываем результаты теста на странице

var resultsDiv = document.getElementById('results');

resultsDiv.style.opacity = '1'; // Делаем результаты видимыми, если они были скрыты

resultsDiv.classList.add('fadeIn'); // Добавляем анимацию для плавного появления результатов

// Выводим результаты теста на странице

var scoreDiv = document.getElementById('score');

scoreDiv.innerHTML = 'Количество правильных ответов: ' + correctAnswers + ' из ' + totalQuestions;

var firstAttemptStatsDiv = document.getElementById('firstAttemptStats');

firstAttemptStatsDiv.innerHTML = 'Ошибки при первых попытках: ' + firstAttemptWrong;

}

2. Функция checkAnswerSelection()

* Эта функция проверяет, был ли выбран хотя бы один ответ на любой из вопросов в викторине. Если ответ выбран, она делает кнопку submitButton видимой для отправки результатов.

// Функция для проверки выбора хотя бы одного ответа

function checkAnswerSelection() {

var anyChecked = false;

// Проверяем каждый вопрос на наличие выбранного ответа

quizQuestions.forEach(function(question) {

var radioButtons = question.querySelectorAll('input[type="radio"]');

radioButtons.forEach(function(radio) {

if (radio.checked) {

anyChecked = true; // Если выбран хотя бы один ответ, устанавливаем true

}

});

});

// Если выбран хотя бы один ответ, показываем кнопку отправки

if (anyChecked) {

submitButton.classList.remove('submitButtonHidden'); // Удаляем класс скрытия кнопки

submitButton.classList.add('submitButtonVisible'); // Добавляем класс видимости кнопки

}

}

3. Обработчики событий на радио-кнопки

* Эти обработчики событий назначают функции calculateScore() и checkAnswerSelection() на событие change для каждой радио-кнопки в каждом вопросе в викторине. При изменении выбора они вызывают соответствующие функции для пересчета результатов и проверки выбора ответа.

// Назначаем обработчики событий на радио-кнопки для каждого вопроса в викторине

quizQuestions.forEach(function(question) {

var radioButtons = question.querySelectorAll('input[type="radio"]');

radioButtons.forEach(function(radio) {

radio.addEventListener('change', function() {

calculateScore(); // Вызываем функцию calculateScore при изменении выбора

checkAnswerSelection(); // Проверяем выбор ответа

});

});

});

4. Анимация вопросов и черной формы

* Этот код добавляет анимации для вопросов в викторине (slideInLeft и slideInRight), задерживая каждую анимацию на 500 миллисекунд для создания эффекта последовательности. Также он добавляет анимацию (fallIn) для элемента quizOpros, который является черной формой. После завершения анимации черной формы, показываются кнопки с помощью добавления класса buttonsVisible к контейнеру кнопок.

// Добавляем анимации для вопросов

quizQuestions.forEach(function(question, index) {

setTimeout(function() {

question.classList.add(index % 2 === 0 ? 'slideInLeft' : 'slideInRight');

}, index \* 500);

});

// Анимация для черной формы

var quizOpros = document.getElementById('quizOpros');

quizOpros.classList.add('fallIn');

quizOpros.addEventListener('animationend', function() {

var buttonContainer = document.getElementById('buttonContainer');

buttonContainer.classList.add('buttonsVisible');

});

5. Общее использование функций showResults() и expandQuizBlock():

* showResults(): Скрывает элемент с тестом (quizOpros) и показывает элемент с результатами (results) с анимацией fadeIn.
* expandQuizBlock(): Перенаправляет пользователя на другую страницу с URL index1.htm.

Этот код реализует функционал викторины, обрабатывая ответы пользователей, подсчитывая результаты, добавляя анимации и управляя видимостью элементов на странице в зависимости от действий пользователя.

# Заключение

В ходе разработки проекта были успешно реализованы все требования к функционалу, которые были установлены на начальном этапе проектирования приложения. Этот процесс позволил приобрести ценные навыки в создании HTML-страниц и работе с JavaScript.

В этом проекте был изучен язык разметки гипертекстовых документов HTML, его основные функции, свойства и параметры. Сегодня HTML применяется во всех без исключения электронных документах, независимо от тематики, величины и коммерческой направленности интернет-проектов. HTML является основополагающим языком для создания веб-страниц и играет ключевую роль в структуре и отображении контента в браузерах.

Существуют многочисленные программы для создания веб-сайтов, однако ни одна из них не может выполнить эту задачу самостоятельно без участия пользователя. Пользователь должен управлять программой, вводить данные и настраивать параметры. Независимо от того, какую программу использовать, знание языка гипертекстовой разметки HTML остается необходимым условием для успешного создания и редактирования веб-страниц. Без этого знания пользователю не удастся добиться желаемого результата.

В процессе создания веб-сайта были освоены навыки работы с файлами различных форматов и расширений для их корректного отображения в интернете. Это включает в себя умение работать с графическими, текстовыми и мультимедийными файлами, а также знание способов их оптимизации для быстрой загрузки и корректного отображения на различных устройствах.

Язык веб-программирования JavaScript в последнее время приобрел большую популярность благодаря своей гибкости и возможности создания интерактивных элементов на веб-страницах. В рамках данного проекта были изучены основы JavaScript, его функции и методы, что позволило добавить интерактивные элементы и улучшить пользовательский опыт на созданных страницах.

Таким образом, разработка проекта стала значительным шагом в изучении HTML и JavaScript, а также в понимании принципов работы современных веб-технологий. Полученные знания и навыки будут полезны в дальнейших проектах и помогут создавать более сложные и функциональные веб-приложения.

# Список литературы

|  |  |
| --- | --- |
| [1] | «Документация Angular,» [В Интернете]. Available: https://angular.io/docs. |
| [2] | Иванов, *Руководство по Ангуляр,* сайт www. angular.com. |
| [3] | Д. Дакетт, HTML и CSS. Разработка и дизайн веб-сайтов.. |
| [4] | Д. Флэнаран, JavaScript: карманный справочник, 2013. |
| [5] | «GSAP,» [В Интернете]. Available: https://gsap.com/. [Дата обращения: 12 Июль 2024]. |
| [6] | Р. Томаль, Бесплатное руководство по Основам Web-дизайна. |
| [7] | «Онлайн учебник JavaScript,» [В Интернете]. Available: https://learn.javascript.ru/. [Дата обращения: 14 Июль 2024]. |
| [8] | П. Рэнд, Дизайн. Форма и хаос. |
| [9] | Б. Хеник, HTML и CSS. Путь к совершенству. |
| [10] | «W3 schools,» [В Интернете]. Available: https://www.w3schools.com/. [Дата обращения: 12 Июль 2024]. |

# Приложение

Файл index1

<!DOCTYPE html>

<html lang="ru">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>Обезьяны</title>

<link rel="stylesheet" href="CSSS.css">

</head>

<body>

<div class="backgroundGif"></div>

<div class="topText" onclick="toggleInfo()">Читать О макаках</div>

<div class="mon1" id="infoBlock">

<div class="monPaj" onclick="swingImage()" onmouseover="toggleImageSize()" onmouseout="resetImageSize()"></div>

<div class="textMAKAK">

<p>

Мака́ки — род приматов из семейства мартышковых, состоящий из 21 вида, большинство из которых обитают в Азии.

</p>

<p><span onclick="toggleAdditionalText()">Подробнее...</span></p>

<div class="additionalText" id="additionalText">

Макаки — приматы средней величины с крепким туловищем и сильными конечностями. Их густая шерсть окрашена в серо-коричневый цвет, однако порою бывает и чёрной. На продолговатой морде волосяной покров отсутствует. У некоторых видов на голове имеются заметные «шапки» либо своеобразные бороды. Важным отличительным признаком является длина хвоста: у маготов он полностью отсутствует, у некоторых видов он куцый, а у остальных почти такой же длинный, как всё туловище. Длина тела у макак достигает 80 см, а масса колеблется от 6 до 15 кг. Самцы в среднем в два раза тяжелее самок.

</div>

</div>

</div>

<div class="mon2" id="quizBlock" onclick="expandQuizBlock()">

<p id="quizText">НАЧАТЬ ВИКТОРИНУ</p>

</div>

<script src="JS\_for\_index.js"></script>

</body>

</html>

Файл CSSS

.body {

background-image: url("5.png");

background-attachment: fixed;

background-position: center;

background-size: cover;

margin: 0;

padding: 0;

display: flex;

flex-direction: column;

align-items: center;

}

.backgroundGif {

position: absolute;

top: 0;

left: 0;

width: 100%;

height: 100%;

background-image: url("background.gif");

background-size: cover;

background-position: center;

z-index: 1;

}

.topText {

margin-top: 5vh;

margin-bottom: 5vh;

text-align: center;

color: white;

font-size: 8em;

cursor: pointer;

user-select: none;

z-index: 2;

}

.topText:hover {

animation: pulseAnimation 1s infinite;

}

.mon1 {

height: 500px;

background-color: rgba(0, 0, 0, 0.5);

width: 80%;

border-radius: 20px;

display: flex;

align-items: center;

margin-top: 22.5vh;

margin-bottom: 5vh;

position: absolute;

left: -100%;

transition: left 0.5s ease;

z-index: 2;

}

.mon2 {

height: 100px;

background-color: rgba(0, 0, 0, 0.5);

width: 30%;

border-radius: 20px;

z-index: 2;

display: flex;

justify-content: center;

align-items: center;

color: white;

position: absolute;

font-size: 2em;

bottom: 50vh;

transition: bottom 0.5s ease;

cursor: pointer;

user-select: none;

}

.mon2:hover #quizText {

animation: pulseAnimation 1s infinite;

}

.mon1.show {

left: 10%;

}

.mon1.show + .mon2 {

bottom: 10vh;

}

.monPaj {

height: 100%;

width: 100%;

max-width: 500px;

background-image: url("1.png");

background-size: cover;

background-position: center;

border-radius: 20px;

transition: transform 0.5s ease;

}

.textMAKAK {

height: 500px;

margin-left: 5%;

margin-right: 5%;

max-width: 850px;

color: white;

font-size: 25px;

}

.textMAKAK span {

cursor: pointer;

user-select: none;

}

.textMAKAK span:hover {

animation: pulseAnimation 1s infinite;

}

@keyframes pulseAnimation {

0% { opacity: 1; }

50% { opacity: 0.5; }

100% { opacity: 1; }

}

.monPaj:hover {

transform: scale(0.8);

}

@keyframes swingAnimation {

0% { transform: rotate(0) scale(0.8); }

25% { transform: rotate(5deg) scale(0.8); }

50% { transform: rotate(-5deg) scale(0.8); }

75% { transform: rotate(3deg) scale(0.8); }

100% { transform: rotate(0) scale(0.8); }

}

.additionalText {

display: none;

color: white;

font-size: 15px;

margin-top: 20px;

text-align: left;

opacity: 0;

}

@keyframes textAnimationShow {

0% { opacity: 0; font-size: 15px; }

100% { opacity: 1; font-size: 25px; }

}

@keyframes textAnimationHide {

0% { opacity: 1; font-size: 25px; }

100% { opacity: 0; font-size: 15px; }

}

Файл JS\_for\_index

var originalTransform = '';

// Функция для уменьшения размера изображения

function toggleImageSize() {

var monPaj = document.querySelector('.monPaj');

originalTransform = monPaj.style.transform; // Сохраняем текущее значение transform

monPaj.style.transform = 'scale(0.8)'; // Устанавливаем масштаб 0.8

}

// Функция для восстановления исходного размера изображения

function resetImageSize() {

var monPaj = document.querySelector('.monPaj');

monPaj.style.transform = originalTransform; // Восстанавливаем исходное значение transform

}

// Функция для анимации качели изображения

function swingImage() {

var monPaj = document.querySelector('.monPaj');

monPaj.style.animation = 'swingAnimation 0.5s ease'; // Запускаем анимацию swingAnimation

setTimeout(function() {

monPaj.style.animation = ''; // Очищаем анимацию после 0.5 секунды

}, 500);

}

// Функция для переключения дополнительного текста

function toggleAdditionalText() {

var additionalText = document.getElementById('additionalText'); // Получаем элемент по его id 'additionalText' и сохраняем его в переменную additionalText

if (additionalText.style.display === 'block') {

// Если текст отображается, скрываем его с анимацией

additionalText.style.animation = 'textAnimationHide 0.5s forwards'; // Применяем анимацию textAnimationHide длительностью 0.5 секунды с заполнением состояния

setTimeout(function() {

additionalText.style.display = 'none'; // Скрываем элемент после завершения анимации через 0.5 секунды

}, 500);

} else {

// Если текст скрыт, показываем его с анимацией

additionalText.style.display = 'block'; // Отображаем элемент

additionalText.style.animation = 'textAnimationShow 0.5s forwards'; // Применяем анимацию textAnimationShow длительностью 0.5 секунды с заполнением состояния

}

}

// Функция для переключения информационного блока

function toggleInfo() {

var infoBlock = document.getElementById('infoBlock'); // Получаем элемент с id 'infoBlock' и сохраняем его в переменную infoBlock

var quizBlock = document.getElementById('quizBlock'); // Получаем элемент с id 'quizBlock' и сохраняем его в переменную quizBlock

if (infoBlock.classList.contains('show')) {

// Если информационный блок показан, скрываем его

infoBlock.classList.remove('show'); // Удаляем класс 'show' у элемента infoBlock, чтобы скрыть блок

quizBlock.style.bottom = '50vh'; // Устанавливаем свойство bottom для quizBlock на 50% высоты окна, чтобы корректно отображать элемент под информационным блоком

} else {

// Если информационный блок скрыт, показываем его

infoBlock.classList.add('show'); // Добавляем класс 'show' элементу infoBlock, чтобы показать блок

quizBlock.style.bottom = '10vh'; // Устанавливаем свойство bottom для quizBlock на 10% высоты окна, чтобы поднять его над информационным блоком

}

}

function expandQuizBlock() {

// Перенаправление на другую страницу

window.location.href = 'test1.htm';

}

Файл test1

<!DOCTYPE html>

<html lang="ru">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>тест</title>

<link rel="stylesheet" href="CSS\_for\_test.css">

</head>

<body>

<div class="quizOpros" id="quizOpros">

<div class="quizQuestion">

<h3>Вопрос 1: К какому семейству относятся макаки?</h3>

<div class="quizAnswers">

<input type="radio" id="q1a1" name="q1" value="a1" data-correct>

<label for="q1a1">Семейство мартышковых</label>

<input type="radio" id="q1a2" name="q1" value="a2">

<label for="q1a2">Семейство кошачьих</label>

<input type="radio" id="q1a3" name="q1" value="a3">

<label for="q1a3">Семейство крыс</label>

</div>

</div>

<div class="quizQuestion">

<h3>Вопрос 2: Где обитают большинство видов макак?</h3>

<div class="quizAnswers">

<input type="radio" id="q2a1" name="q2" value="a1" data-correct>

<label for="q2a1">В Азии</label>

<input type="radio" id="q2a2" name="q2" value="a2">

<label for="q2a2">В Африке</label>

<input type="radio" id="q2a3" name="q2" value="a3">

<label for="q2a3">В Южной Америке</label>

</div>

</div>

<div class="quizQuestion">

<h3>Вопрос 3: Каковы основные отличительные черты макак?</h3>

<div class="quizAnswers">

<input type="radio" id="q3a1" name="q3" value="a1">

<label for="q3a1">Густая шерсть и длинный хвост</label>

<input type="radio" id="q3a2" name="q3" value="a2">

<label for="q3a2">Отсутствие хвоста и чёрный цвет шерсти</label>

<input type="radio" id="q3a3" name="q3" value="a3" data-correct>

<label for="q3a3">Заметные "шапки" на голове и длинные уши</label>

</div>

</div>

</div>

<div class="buttonContainer" id="buttonContainer">

<button class="quizButton" id="backButton" onclick="expandQuizBlock()">Назад</button>

<button class="quizButton submitButtonHidden" id="submitButton" onclick="showResults()">Отправить</button>

</div>

<div class="results" id="results">

<h2>Результаты теста</h2>

<div id="score"></div>

<div id="firstAttemptStats"></div>

</div>

<script src="JS\_for\_test.js"></script>

</body>

</html>

Файл CSS\_for\_test

body {

background-image: url("5.png");

background-attachment: fixed;

background-position: center;

background-size: cover;

margin: 0;

padding: 0;

display: flex;

flex-direction: column;

align-items: center;

height: 100vh;

overflow: hidden;

}

.backgroundGif {

position: absolute;

top: 0;

left: 0;

width: 100%;

height: 100%;

background-image: url("background.gif");

background-size: cover;

background-position: center;

z-index: 1;

}

.quizOpros {

position: absolute;

margin-bottom: 20px;

top: 50%;

left: 50%;

transform: translate(-50%, -50%);

background-color: rgba(0, 0, 0, 0.7);

width: 50%;

padding: 20px;

border-radius: 10px;

box-shadow: 0 4px 8px rgba(0, 0, 0, 0.2);

color: white;

font-size: 1.5em;

margin-bottom: 10px;

opacity: 0;

}

.quizAnswers {

display: flex;

flex-direction: column;

}

.quizAnswers label {

background-color: rgba(255, 255, 255, 0.1);

border-radius: 10px;

padding: 10px 20px;

margin-bottom: 10px;

cursor: pointer;

transition: background-color 0.3s ease;

}

.quizAnswers label:hover {

background-color: rgba(255, 255, 255, 0.3);

}

.quizAnswers input[type="radio"] {

display: none;

}

.quizAnswers input[type="radio"]:checked + label {

background-color: rgba(0, 255, 0, 0.5); /\* Зеленая подсветка для правильного ответа \*/

}

.quizAnswers input[type="radio"]:checked:not([data-correct]) + label {

background-color: rgba(255, 0, 0, 0.5); /\* Красная подсветка для неправильного ответа \*/

}

.quizQuestion {

opacity: 0;

position: relative;

}

.slideInLeft {

animation: slideInLeft 1s forwards;

}

.slideInRight {

animation: slideInRight 1s forwards;

}

@keyframes slideInLeft {

from {

left: -100%;

opacity: 0;

}

to {

left: 0;

opacity: 1;

}

}

@keyframes slideInRight {

from {

right: -100%;

opacity: 0;

}

to {

right: 0;

opacity: 1;

}

}

.fallIn {

animation: fallIn 1s ease-out forwards;

}

@keyframes fallIn {

0% {

top: -100%;

opacity: 0;

}

70% {

top: 50%;

opacity: 1;

}

90% {

top: 45%;

opacity: 1;

}

100% {

top: 50%;

opacity: 1;

}

}

.buttonContainer {

position: absolute;

top: 10%;

width: 80%;

display: flex;

justify-content: space-between;

padding: 0 20px;

box-sizing: border-box;

opacity: 0;

transition: opacity 1s ease;

}

.quizButton {

background-color: rgba(0, 0, 0, 0.7);

color: white;

border: none;

padding: 10px 20px;

font-size: 1em;

border-radius: 5px;

cursor: pointer;

transition: background-color 0.3s ease;

}

.quizButton:hover {

background-color: rgba(0, 0, 0, 0.9);

}

.buttonsVisible {

opacity: 1;

}

.submitButtonVisible {

display: block;

}

.submitButtonHidden {

display: none;

}

.results {

display: none;

position: absolute;

top: 50%;

left: 50%;

transform: translate(-50%, -50%);

text-align: center;

padding: 20px;

background-color: rgba(0, 0, 0, 0.7);

border-radius: 10px;

color: white;

font-size: 1.5em;

width: 50%;

opacity: 0;

transition: opacity 1s ease;

}

@keyframes fadeIn {

from {

opacity: 0;

}

to {

opacity: 1;

}

}

Файл JS\_for\_test

// Обработчик события при загрузке DOM

document.addEventListener('DOMContentLoaded', function() {

var quizQuestions = document.querySelectorAll('.quizQuestion');

var firstAttemptCorrect = 0;

var firstAttemptWrong = 0;

var submitButton = document.getElementById('submitButton');

// Функция для подсчета правильных и неправильных ответов

function calculateScore() {

var totalQuestions = quizQuestions.length;

var correctAnswers = 0;

quizQuestions.forEach(function(question) {

var radioButtons = question.querySelectorAll('input[type="radio"]');

var isCorrect = false;

var isFirstAttempt = true;

radioButtons.forEach(function(radio) {

if (radio.checked) {

var label = radio.nextElementSibling;

if (radio.hasAttribute('data-correct')) {

label.style.backgroundColor = 'rgba(0, 255, 0, 0.5)'; // Правильный ответ

correctAnswers++;

isCorrect = true;

} else {

label.style.backgroundColor = 'rgba(255, 0, 0, 0.5)'; // Неправильный ответ

if (isFirstAttempt) {

firstAttemptWrong++;

}

}

isFirstAttempt = false; // После первого выбора

}

});

// Если правильный ответ выбран, отключаем все радио-кнопки

if (isCorrect) {

radioButtons.forEach(function(radio) {

radio.disabled = true;

});

}

});

// Выводим результаты в консоль (или можно вставить другой код для отображения результата)

console.log('Количество правильных ответов: ' + correctAnswers + ' из ' + totalQuestions);

console.log('Ошибки при первых попытках: ' + firstAttemptWrong);

// Показываем результаты теста

var resultsDiv = document.getElementById('results');

resultsDiv.style.opacity = '1'; // Показываем результаты, если они были скрыты

resultsDiv.classList.add('fadeIn'); // Добавляем анимацию для плавного появления результатов

// Выводим результаты теста

var scoreDiv = document.getElementById('score');

scoreDiv.innerHTML = 'Количество правильных ответов: ' + correctAnswers + ' из ' + totalQuestions;

var firstAttemptStatsDiv = document.getElementById('firstAttemptStats');

firstAttemptStatsDiv.innerHTML = 'Ошибки при первых попытках: ' + firstAttemptWrong;

}

// Функция для проверки выбора хотя бы одного ответа

function checkAnswerSelection() {

var anyChecked = false;

quizQuestions.forEach(function(question) {

var radioButtons = question.querySelectorAll('input[type="radio"]');

radioButtons.forEach(function(radio) {

if (radio.checked) {

anyChecked = true;

}

});

});

if (anyChecked) {

submitButton.classList.remove('submitButtonHidden');

submitButton.classList.add('submitButtonVisible');

}

}

// Назначаем обработчики событий на радио-кнопки для каждого вопроса в викторине

quizQuestions.forEach(function(question) {

var radioButtons = question.querySelectorAll('input[type="radio"]');

radioButtons.forEach(function(radio) {

radio.addEventListener('change', function() {

calculateScore(); // Вызываем функцию calculateScore при изменении выбора

checkAnswerSelection(); // Проверяем выбор ответа

});

});

});

// Добавляем анимации для вопросов

quizQuestions.forEach(function(question, index) {

setTimeout(function() {

question.classList.add(index % 2 === 0 ? 'slideInLeft' : 'slideInRight');

}, index \* 500);

});

// Анимация для черной формы

var quizOpros = document.getElementById('quizOpros');

quizOpros.classList.add('fallIn');

// Показать кнопки после завершения анимации формы

quizOpros.addEventListener('animationend', function() {

var buttonContainer = document.getElementById('buttonContainer');

buttonContainer.classList.add('buttonsVisible');

});

});

function showResults() {

var quizOpros = document.getElementById('quizOpros');

var results = document.getElementById('results');

// Скрываем тест

quizOpros.style.display = 'none';

// Показываем результаты

results.style.display = 'block';

results.classList.add('fadeIn'); // Добавляем анимацию для появления результатов

}

function expandQuizBlock() {

// Перенаправление на другую страницу

window.location.href = 'index1.htm'; // Замените 'путь-к-вашей-странице' на реальный URL

}