специальности и составляет 16 или 32 часа. Нами принят вариативный подход к формированию конкретного набора работ для каждой группы студентов. Такой подход позволяет повысить самостоятельность и мотивировать студентов. Практикум включает более тридцати пяти работ. Основными разделами лабораторного практикума являются: «Дизайн и события мобильных приложений», «Данные в мобильных приложениях», «Мобильные приложения для управления базами данных SQLite», «Файловая система ОС Android», «Мобильные приложения для обмена данными и их обработки», «Мобильные приложения с использованием сенсоров», «Геолокация», «Мобильные приложения с подключением внешних модулей».

Большой перечень разделов и разнообразие лабораторных работ дает возможность индивидуализации обучения на основе вариативного подхода. При этом конкретный набор для студента может быть максимально приближен к его интересам в направлении курсового и дипломного проектирования.

Приведем пример индивидуального набора лабораторных работ практикума, соответствующих студенческому исследованию по теме «Разработка мобильного приложения распознавания голосов птиц».

Лабораторная работа по теме «Разработка приложений с несколькими Activity. Передача данных между Activity», целью которой является формирование представлений о структуре мобильных приложений, взаимодействии форм и активностей.

Лабораторная работа по теме «Списки. Создание собственного адаптера. Механизмы обратного вызова», содержание которой посвящено созданию адаптеров, позволяющих визуализировать структурированную информацию и возможности выбора компонентов.

Лабораторная работа «Работа с локальной базой данных» направлена на формирование компетенций для работы с SQLite базами данных.

Лабораторная работа «Взаимодействие с сервером. Потоки» имеет целью формирование компетенций, необходимых для создания приложений, взаимодействующих с серверными хранилищами.

Лабораторная работа «Хранение данных. Настройки и внешние файлы» позволяет получить дополнительные знания и умения, необходимые для использования файловой системы Android.

Лабораторная работа «Приложение для воспроизведения аудио- и видео файлов» знакомит со средствами обработки мультимедийной информации.

Лабораторная работа «Мобильное приложение для обработки облачной базы данных» направлена на формирование умений, необходимых для создания приложений, взаимодействующих с облачными хранилищами.

Лабораторная работа «Мобильное приложение для решения задач с использованием нейронной сети» играет важную роль в формировании представлений о подключении внешних модулей, в частности, готовых нейронных сетей для распознавания графики или звука.

Список использованных источников

1. Самые популярные операционные системы в мире основаны на ядре Linux [Электронный ресурс]. – Режим доступа: https://luckyea77.livejournal.com/4440943.html. – Дата доступа: 31.01.2024.

2. ОСВО 1-53 01 02-202 для специальности 1-53 01 02 Автоматизированные системы обработки информации [Электронный ресурс]. – Режим доступа: http://asu.bru.by/кафедра/Учебно-методические материалы/Учебные планы/ 1-53 01 02-202 АСОИ-2021/Проект ОСВО 201-53-01-02_150421.pdf. – Дата доступа: 31.01.2024.

3. Разработка приложений для мобильной операционной системы (на примере ОС «ANDROID») [Электронный ресурс]. – Режим доступа: https://rep.brsu.by/handle/123456789/417. – Дата доступа: 31.01.2024.

4. Уроки по Android [Электронный ресурс]. – Режим доступа: https://startandroid.ru/ru/. – Дата доступа: 31.01.2024.

5. Программирование под Андроид на Java [Электронный ресурс]. – Режим доступа: https://metanit.com/java/android/. – Дата доступа: 31.01.2024.

УДК 004.432.2

И. А. КОЛЕСНИКОВ, А. А. ГОЛУБ, А. П. САФРОНОВ

УО «Мозырский государственный педагогический университет им. И. П. Шамякина» (г. Мозырь, Беларусь)

РАБОТА С ПАМЯТЬЮ БРАУЗЕРА. ЗАГРУЗКА ИЗОБРАЖЕНИЙ И РАБОТА С НИМИ

Веб-разработка не ограничивается лишь созданием собственных красивых и практичных пользовательских интерфейсов. Зачастую пользователю необходимо загружать файлы напрямую из компьютера в память браузера с возможностью в дальнейшем использовать их в личных нуждах [1].

В рамках данной лабораторной работы разработаем сайт, который хранит загруженные изображения в памяти браузера, отображает на экране список ранее загруженных изображений и позволяет пользователю взаимодействовать с ними.

Первым делом нам необходимо добавить на страницу следующие элементы:

- загрузку изображений;

• кнопку «Загрузить»;
• список загруженных изображений;
• контейнер для отображения выбранного изображения.

Для удобства разместим все элементы, связанные с загрузкой изображений по центру страницы, а те, что связаны с последующим отображением на странице, поместим в один div контейнер.

```
<center>
<input type="file" id="fileInput" />
<button id="loadButton">Загрузить</button>
</center>
<div class="container">
<div id="imageList"></div>
<div id="imageContainer"></div>
</div>
```

Далее необходимо добавить стили CSS для красивого оформления созданных нами элементов и реализовать JavaScript код для хранения загруженных нами изображений и работы с ними.

Создадим пустой массив images, в котором будут храниться загруженные изображения:

```
var images = [];
```

Далее нам нужно добавить обработчик события на кнопку «Загрузить» [2]:

```
document.getElementById("loadButton").addEventListener("click", function () { });
```

При клике на кнопку будет выполняться функция, которая начнет процесс загрузки изображения [2].

Поместим внутрь фигурных скобок основной код для работы с кнопкой «Загрузить»:

```
var fileInput = document.getElementById("fileInput");
var file = fileInput.files[0];
if (file && file.type.startsWith("image/")) {
var reader = new FileReader();
reader.onload = function (e) {
var image = new Image();
image.src = e.target.result;
images.push(image);
renderImageList();          };
reader.readAsDataURL(file);
} else { alert("Пожалуйста, выберите файл изображения.");}
```

В указанном выше фрагменте реализовано получение доступа к элементу input с id «fileInput», в котором пользователь выбирает файл. Далее мы получаем первый выбранный файл из элемента input и проверяем, что файл существует и его тип начинается «image/», что в дальнейшем избавит нас от случайной загрузки неверного формата данных. Создаем объект FileReader для чтения содержимого файла и устанавливаем обработчик, который будет вызван при успешном завершении чтения файла. Для отображения уже загруженного пользователем изображения необходимо создать новый элемент Image, задать путь к загруженному изображению в его свойство src и добавить этот элемент в массив, хранящий в себе всё изображения images. Вызовем функцию для отображения списка изображений на странице renderImageList(). Запускаем операцию чтения содержимого в файле в формате Data URL, по окончанi которой будет вызван обработчик onload. Команда «else { alert("Пожалуйста, выберите файл изображения.");}» Нужна для вывода предупреждения пользователю через alert.

Теперь необходимо создать функцию «renderImageList()», которую мы уже ранее вызвали в коде.

```
    function renderImageList() {
var imageList = document.getElementById("imageList");
    imageList.innerHTML = "";
    images.forEach(function (image, index) {
    var imgElement = document.createElement("img");
    imgElement.src = image.src;
imgElement.addEventListener("click", function ()  {displaySelectedImage(index);  });
     imageList.appendChild(imgElement);   });   }
```

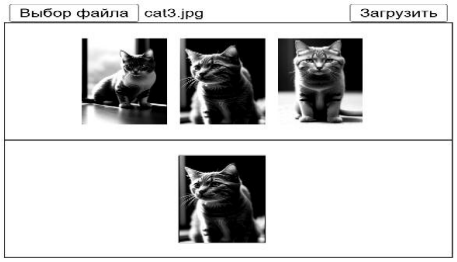
Разбирая её по порядку, увидим: получение доступа к элементу imageList; очищение его; обращение к функции обратного вызова, принимающей текущий элемент image и его индекс index; создание нового элемента imgElement, представляющий отдельное изображение в списке; установление пути к нему из массива images; добавление обработчика события при выборе изображения, в котором вызывается функция displaySelectedImage с передачей индекса в качестве аргумента; добавление созданного элемента в контейнер imageList [2]. Таким образом, каждое изображение будет отображаться на странице.

В заключении необходимо создать функцию «displaySelectedImage». Как было сказано выше, эта функция будет принимать индекс в качестве аргумента:

```
function displaySelectedImage(index) {       var imageContainer =
```

```
document.getElementById("imageContainer");
        imageContainer.innerHTML = "";
        var image = images[index];
        var imgElement = document.createElement("img");
        imgElement.src = image.src;
        imgElement.classList.add("selected");
        imageContainer.appendChild(imgElement);     }
```

Она позволяет получить доступ к соответствующему изображению из созданного ранее массива. После его получения создаем новый элемент и устанавливает путь к выбранному изображению в качестве его источника. Затем к созданному элементу добавляется класс "selected". Далее функция добавляет его



в контейнер на странице, который представляет собой область для отображения выбранного изображения.

Рисунок 1 – Пример готовой страницы

Как итог (рисунок 1), весь JavaScript код работает с массивом images, который хранит загруженные изображения и использует его для отображения списка уже полученных ранее элементов. Информация о загруженных изображениях сохраняется в памяти браузера в виде объектов Image, что позволяет управлять ими и отображать на странице.

Список использованных источников

1. Хавербеке, Марейн. Выразительный Javascript / Марейн Хавербеке. – 2-е изд. – СПб. : Питер, 2015. – 425 с.

2. Макфарланд, Дэвид. Javascript и jQuery : исчерпывающее руководство / Дэвид Макфарланд. – 3-е изд. – М. : Эксмо, 2015. – 880 с.

УДК 37.03

О. В. КОРЕЙБА

ФГБОУ ВО «Армавирский государственный педагогический университет» (г. Армавир, Россия)

ОРГАНИЗАЦИЯ РАБОТЫ С ОДАРЕННЫМИ ДЕТЬМИ В ДОПОЛНИТЕЛЬНОМ ОБРАЗОВАНИИ

Сегодня в ситуации глобальных преобразований всех сфер жизни российского общества вновь актуализировались вопросы детской одаренности. Экономика нашей страны остро нуждается в специалистах, обладающих глубокими знаниями, нестандартным мышлением, творческой активностью и высокой социальной ответственностью. Именно талантливые школьники, демонстрирующие незаурядные способности, представляют для государства основной интерес в качестве интеллектуального и творческого потенциала, обуславливающего будущее развитие страны.

Социальный заказ на активизацию педагогических усилий по выявлению и развитию молодых талантов отчетливо выражен в нормативных документах сферы образования последнего десятилетия, основными из которых являются: федеральная целевая программа развития образования на 2011–2015 годы (утверждена постановлением Правительства РФ от 7 февраля 2011 г. № 61), национальная стратегия действий в интересах детей Российской Федерации до 2017 года (утверждена указом Президента РФ от 1 июня 2012 г. № 761), государственная программа Российской Федерации «Развитие образования» на 2013 – 2020 годы (утверждена распоряжением Правительства РФ от 22 ноября 2012 г. № 2148-р), концепция Российской национальной системы выявления и развития молодых талантов (утверждена Президентом РФ 3 апреля 2012 г.).

Одаренность – это системное, развивающееся в течение жизни качество психики, которое определяет возможность достижения человеком более высоких, незаурядных результатов в одном или нескольких видах деятельности по сравнению с другими людьми [1].

Одаренный ребенок – это ребенок, который выделяется яркими, очевидными, иногда выдающимися достижениями (или имеет внутренние предпосылки для таких достижений) в том или ином виде деятельности.

Целью образования (обучения и воспитания) одаренных детей является формирование знаний, умений и навыков в определенных областях, а также создание условий для познавательного и личностного развития учащихся с учетом их дарования.