

4 ЛАБОРАТОРНАЯ РАБОТА №4

«Исследование возможностей библиотеки jQuery»

4.1 Цель работы

Исследовать эффективность применения библиотек при разработке клиентских приложений на примере библиотеки jQuery. Изучить возможность программирования на клиентской стороне с использованием библиотеки jQuery. Приобрести практические навыки использования библиотеки jQuery для обработки форм, модификации содержимого HTML-страницы, создания эффектов анимации.

4.3 Ход выполнения работы

4.3.1 В начале выполнения лабораторной работы был модифицирован весь JavaScript-код с помощью библиотеки jQuery. После этого была модифицирована страница альбома, была добавлена возможность просмотра увеличенных версий изображений с анимацией. Код файла, содержащего код для этого представлен в листинге 4.1.

Листинг 4.1 – Код для перелистывания изображений

```
$(document).ready(function() {  
    let formIsClosed = true;  
    let currentImgIndex = 0;  
    const images = $('img');  
    const form = $('<div></div>');  
  
    images.each(function(index) {  
        $(this).on('click', function(event) {  
            if (formIsClosed) {  
                currentImgIndex = index;  
                openPhoto(event);  
            }  
        })  
    })  
})
```

```

    });
  });

function openPhoto(event) {
  if (formIsClosed) {
    $('.main-body').css({
      'filter': 'blur(5px)',
      'transition': '1s'
    });
    addPhotoForm(event.target);
  }
}

function addPhotoForm(targetImg) {
  form.addClass('photoForm').css('display', 'block');
  $('body').append(form);
  form.css('top', `${window.scrollY + 100}px`);

  const img = $('<img>').attr({
    src: `img/${getCharacters(targetImg.src)}`,
    alt: targetImg.alt,
    title: targetImg.alt
  }).css({
    'width': '550px',
    'maxHeight': '550px',
    'marginTop': '10px'
  });
  form.append(img);

  $('body').css('overflow', 'hidden');

  const photoDiv = $('<div></div>').addClass('photoDiv').css({
    'display': 'flex',
    'flex-wrap': 'wrap',
    'justify-content': 'center',
    'align-items': 'center',

```

```

        'margin-top': '25px'
    });

    const photoCounter = $('<p></p>').text(`photo
    ${$(targetImg).attr('id')} из ${images.length}`);

    const leftArrow = $('<button></button>').addClass('form-
    but').text('<').css({
        'margin': '0 20px',
        'width': '20%'
    });

    const rightArrow = $('<button></button>').addClass('form-
    but').text('>').css({
        'margin': '0 20px',
        'width': '20%'
    });

    const closeImg = $('<img>').attr({
        src: "img/close.png",
        alt: "close",
        title: "Заккрыть"
    }).addClass('form-but').css({
        'width': '50px',
        'background': '#fff',
        'position': 'absolute',
        'top': '-2%',
        'right': '2%'
    });

    closeImg.on('click', function() {
        closePhotoForm(img, closeImg, photoDiv);
    });

    leftArrow.on('click', function() {
        changePhoto('left', img, photoCounter);
    });

    rightArrow.on('click', function() {
        changePhoto('right', img, photoCounter);
    });

```

```

    photoDiv.append(leftArrow);
    photoDiv.append(photoCounter);
    photoDiv.append(rightArrow);
    form.append(closeImg);
    form.append(photoDiv);

    formIsClosed = false;
}

function changePhoto(direction, img, photoCounter) {
    if (direction === 'left') {
        currentImgIndex = (currentImgIndex === 0) ? images.length-1
: currentImgIndex-1;
    } else {
        currentImgIndex = (currentImgIndex === images.length-1) ? 0
: currentImgIndex+1;
    }

    const newImg = images.eq(currentImgIndex);
    img.fadeOut(300, function() {
        img.attr({
            'src': `img/${getCharacters(newImg.attr('src'))}`,
            'alt': newImg.attr('alt'),
            'title': newImg.attr('alt')
        });
        img.fadeIn(300);
    });

    photoCounter.text(`фото ${newImg.attr('id')} из 15`);
}

function closePhotoForm(img, closeImg, photoDiv) {
    photoDiv.remove();
    form.remove();
    img.remove();
}

```

```

closeImg.remove();

$('body').css('overflow', 'auto');

$('.main-body').css('filter', 'none');

formIsClosed = true;
}

function getCharacters(str) {
    const position = str.lastIndexOf("/");
    if (position === -1) return '';
    return str.substring(position + 1);
}

});

```

На рисунке 4.1 можно увидеть готовый результат открывающихся изображений.

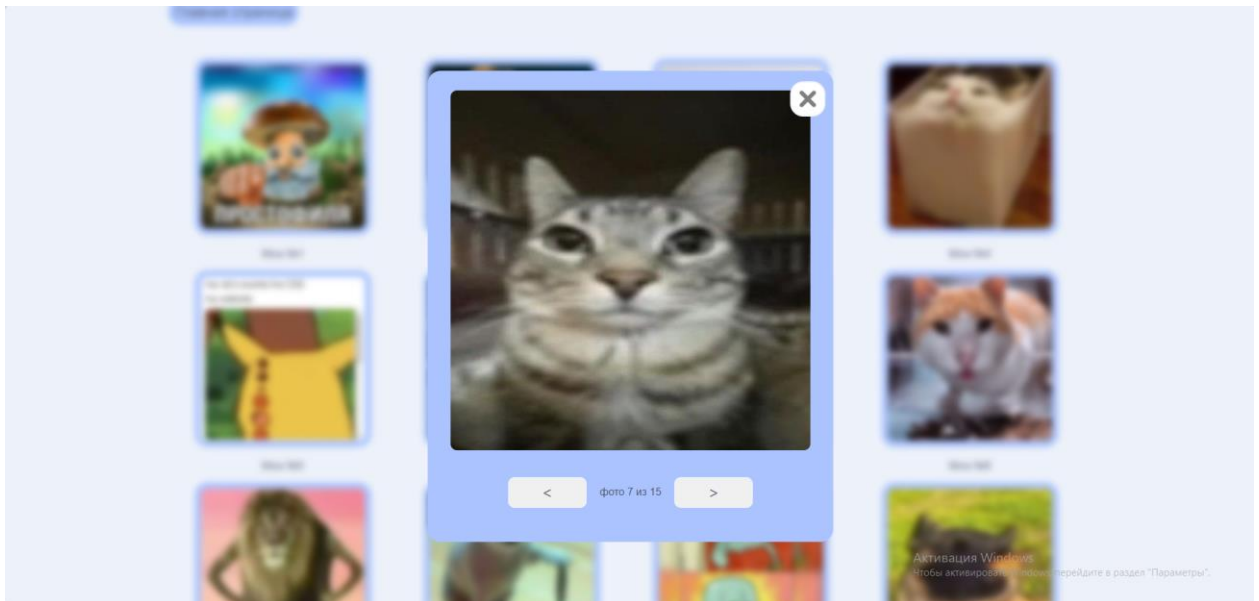


Рисунок 4.1 – Открывающееся изображение

4.3.2 Далее была реализована возможность отображения всплывающего блока, который показывается рядом с элементом, если на него наводится указатель мыши и который исчезает через 1 секунду после того, как указатель мыши покидает элемент. Код файла, содержащего функции для этого представлен в листинге 4.3.

Листинг 4.2 – Код отображения всплывающего блока

```
const fields = ['fio', 'birthday', 'email', 'phone', 'message'];

fields.forEach(field => {

    const inputElement = contactForm.find(`[name="${field}"]`);

    if (inputElement.length) {

        inputElement.on('blur', () => validateField(field));

        inputElement.on('input', () => validateField(field));

        inputElement.on('mouseenter', () => showPopover(field));

        inputElement.on('mouseleave', () => hidePopover(field));

    }

});

function showPopover(fieldName) {

    const field = contactForm.find(`[name="${fieldName}"]`);

    const popover = $('#${fieldName}-popover');

    const offset = field.offset();

    popover.css({

        top: offset.top,

        left: offset.left + field.outerWidth() + 10,

    }).show();

}

function hidePopover(fieldName) {

    const popover = $('#${fieldName}-popover');

    setTimeout(() => popover.hide(), 1000);

}
```

Рисунок 4.2 демонстрирует, как выглядят всплывающие блоки.

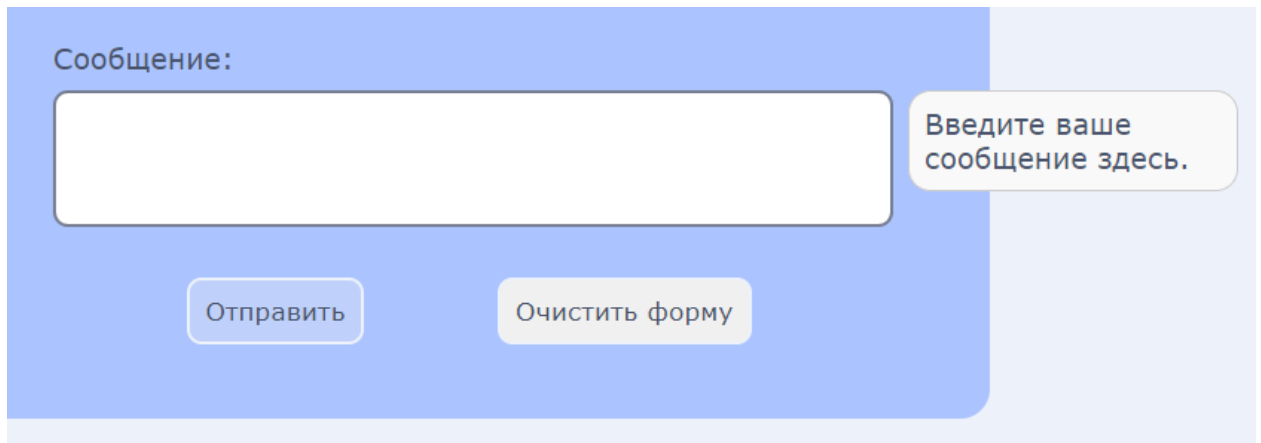


Рисунок 4.2 – Всплывающий блок

4.3.3 Последней была реализована возможность отображения всплывающего модального окна, которое показывается в центре экрана. Код файла, содержащего функции для этого представлен в листинге 4.3.

Листинг 4.3 – Функции модального окна

```
modalWindow();

function modalWindow() {
    const form = $("#contactForm");
    $("#reset-btn").on("click", function(e) {
        e.preventDefault();

        $(".modal-window").remove();
        $(".main-body, .nav-div").addClass("blur-background");

        const window = $('<div></div>').addClass('modal-window');
        const windowText = $('<p></p>').text("Вы уверены, что хотите
очистить форму?").css({
            'font-size': '20px',
        });
        const buttonYes = $('<button></button>').text("Да").addClass('form-but').css({
            'float': 'right',
            'margin-right': '60px',
            'width': '20%',
```

```

    });

    const buttonNo =
    $('<button></button>').text("Нет").addClass('form-but').css({
        'margin-left': '60px',
        'width': '20%',
    });

    buttonYes.on('click', function(){
        resetForm(contactForm);
        window.remove();
        $(".main-body, .nav-div").removeClass("blur-
background");
    });
    buttonNo.on('click', function(){
        window.remove();
        $(".main-body, .nav-div").removeClass("blur-
background");
    });

    window.append(windowText);
    window.append(buttonYes);
    window.append(buttonNo);
    $('body').append(window);

    window.show();
    });
}

```

Рисунок 4.3 демонстрирует внешний вид всплывающего окна.

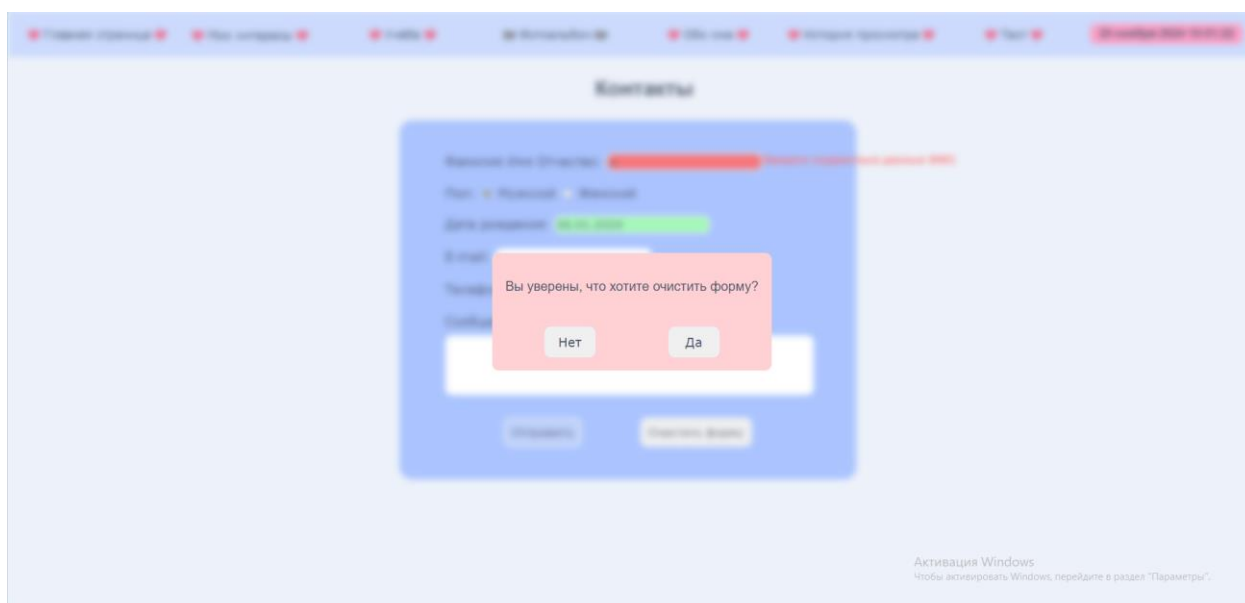


Рисунок 4.3 – Внешний вид всплывающего окна

Выводы

В ходе лабораторной работе была исследована эффективность применения библиотек при разработке клиентских приложений на примере библиотеки jQuery. Также изучена возможность программирования на клиентской стороне с использованием библиотеки jQuery. Далее были приобретены практические навыки использования библиотеки jQuery для обработки форм, модификации содержимого HTML-страницы, создания эффектов анимации. В конце выполнения лабораторной работы был написан отчет.