Лабораторная работа №5.
ДИНАМИЧЕСКИЕ WEB-СТРАНИЦЫ. ОБРАБОТКА СОБЫТИЙ.

Цель работы.

Изучить принципы разработки динамических web-страниц с использований клиентских сценариев JavaScript и объектной модели документа. Научиться создавать кроссбраузерные html-страницы. Познакомиться с уровнями и особенностями API и обработкой событий DOM различных уровней.

Задание.

Согласно варианту разработать сценарий JavaScript, решающий одну из задач. Все обработчики событий должны задаваться/удаляться динамически (методы *addEventListener*/ *removeEventListener* для браузеров, поддерживающих спецификацию W3C, и *attachEvent*/ *detachEvent* для браузера IE).

1. Создать два поля ввода на некоторой html-странице. В первом задаётся Id одного из элементов текущей страницы, во втором – строка с описанием любой 2D-трансформации или CSS3-анимации (предполагается, что объявление анимации с помощью правила @keyframes уже присутствует в таблицах стилей, т.е. её имя известно) с параметрами. Разработать скрипт, позволяющий увидеть результат применения указанного во 2-ом поле ввода эффекта при щелчке мыши на элементе, Id которого задан в 1-ом поле ввода. При этом эффекты, задаваемые в предыдущие моменты времени, не должны отображаться. Проверку на правильность ввода эффекта и его параметров производить не нужно. Предполагается, что строка вводится корректно.
2. Создать два поля ввода на некоторой html-страницы. В первом задаётся Id изменяемого элемента, во втором – строка, содержащая html-код, на которую будет произведена замена элементов с указанным Id. Требуемая подмена с сохранением Id элемента должна происходить при клике мыши на элементе, Id которого задан в 1-ом поле ввода. Предыдущие замены запоминать не нужно. После «подмены» элемент должен сохранить свой идентификатор и всё стилевое оформление.
3. Вывести время в правом нижнем углу окна браузера вне зависимости от скроллинга. При нажатии на время, к этому элементу применяется любая из 2D-трансформаций, выбранная случайно, с параметрами, заданными случайно.
4. В новом окне вывести все элементы, являющиеся непосредственными потомками элемента с указанным в поле ввода Id. Открытие нового окна и вывод потомков должен осуществляться при щелчке на элементе, Id которого задан. Формат вывода информации о потомках следующий:

   Id родительского элемента, имя тега родительского элемента, кол-во **непосредственных** потомков:
   1. Имя тега 1-ого потомка или его содержимое (для текстовых узлов);
   2. Имя тега 2-ого потомка или его содержимое (для текстовых узлов);
   3. Имя тега 3-ого потомка или его содержимое (для текстовых узлов);

   и т.д.
5. Связать со всеми элементами страницы обработчик события *Click*, который бы перемещал данный элемент со всеми его потомками во второе окно. Для всех элементов второго окна также регистрируются подобные обработчики события *Click*.
6. Используя DOM, сформировать таблицу с указанным количеством строк и столбцов, после некоторого элемента, с указанным в поле ввода Id. Добавление таблицы в текущий документ должно осуществляться при щелчке на элементе, Id которого задан в поле ввода. Новой таблице присвоить идентификатор вида «*tablHHMM*», где HH – час, MM – минуты момента создания таблицы.
7. Для указанного с помощью Id элемента отобразить поддерево **всех** его потомков в новом окне браузера. Предусмотреть возможность ввода многих Id элементов, расположенных на странице. Открытие нового окна и вывод потомков элемента должен осуществляться только при

щелчке мышью на элементе, Id которого присутствует в поле ввода. Формат ввода Id элементов произвольный. Привязка/удаление обработчиков события *onClick* элементов, Id которых задал пользователь, реализовать по нажатию на кнопку «Задать обработчики onClick».

8. Связать со всеми элементами страницы обработчик события *onMouseOver*, который бы применял к текущему элементу стилевое правило, заданное в отдельном поле ввода. При этом заданное ранее стилевое оформление не должно «теряться». Изначально никаких стилевых правил на странице нет. Контроль правильности ввода нового стилевого правила производить не нужно, предполагается, что в этой строке нет ошибок. Предусмотреть сброс всего оформления любого из элементов страницы по событию *onMouseOver*. Список стилевых правил для оформления: *font-size, font-weight, text-decoration, text-shadow, border-width, color, background-color*. Параметры стилевых правил должны задаваться случайным образом.
9. Создать на странице кнопку, которая бы «убегала» от курсора мыши. Элемент, который в следующий момент времени станет «родителем» такой кнопки, должен выбираться случайно. Кандидатами в «родители» являются все элементы страницы, у которых есть идентификаторы. В любой момент времени на странице должна быть одна «убегающая» кнопка.
10. Связать со всеми элементами страницы обработчик события *Click*, который бы копировал данный элемент со всеми его потомками и вставлял копию в начало страницы, и *DblClick*, который бы удалял текущий элемент. Для нового элемента и всех его потомков также регистрируются обработчики указанных событий.

Порядок выполнения.

1. Изучить теоретические сведения по API DOM.
2. Разработать алгоритм решения задачи.
3. Написать необходимые сценарии JavaScript.
4. Проверить правильность работы скриптов различных браузерах, в том числе мобильных.

Содержание отчета.

1. Титульный лист.
2. Цель работы.
3. Задание.
4. Описание алгоритма решения поставленной задачи.
5. ~~Код разработанных скриптов.~~
6. Результаты работы.
7. Выводы по проделанной лабораторной работе.