Министерство образования Белгородской области
Областное государственное автономное
профессиональное образовательное учреждение
«Белгородский индустриальный колледж»

Рассмотрено
предметно-цикловой комиссией
Протокол заседания № ____
От «____»_____ 2022
Председатель цикловой комиссии
_____/ Третьяк И.Ю.

Методические рекомендации

по выполнению самостоятельной работы по **МДК 08.01**
Проектирование и разработка интерфейсов пользователя
для специальности

09.02.07 Информационные системы и программирование

Разработчик:

Солдатенко М.Н. преподаватель
специальных дисциплин БИК

Белгород 2022

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

МДК 08.01 Проектирование и разработка интерфейсов пользователя профессионального модуля «Разработка дизайна веб-приложений» является специальным, формирующим базовые умения и компетенции для получения выпускником профессиональных умений и компетенций.

Методические рекомендации по выполнению самостоятельной работы студентов специальности 09.02.07 Информационные системы и программирование разработаны в соответствии с рабочей ПМ.08 Разработка дизайна веб-приложений в части МДК 08.01 Проектирование и разработка интерфейсов пользователя, соответствуют требованиям Федерального государственного образовательного стандарта по специальностям среднего профессионального образования.

Методические рекомендации по выполнению самостоятельной работы студентов содержат информацию о том, сколько и какие темы выносятся на самостоятельное изучение, основную и дополнительную литературу, вопросы для самопроверки.

Целью методических рекомендаций по выполнению самостоятельной работы студентов является организация и управление самостоятельной работой студентов в процессе изучения данного МДК.

Форму самостоятельной работы студент выбирает согласно рабочей программе (реферат, презентация, решение задач). К каждой теме предложен план, вопросы проверки и самопроверки.

Методические рекомендации предназначены для студентов очной формы обучения специальности 09.02.07 Информационные системы и программирование. По учебному плану по МДК 08.01 Проектирование и разработка интерфейсов пользователя на самостоятельную работу студентов отводится 4 часа, на консультации - 12 часов.

Выполненная работа позволит приобрести не только знания, но и умения, навыки, компетенции, а также поможет выработать свою методику подготовки, что очень важно в дальнейшем процессе обучения.

ОБЩИЕ ПОЛОЖЕНИЯ

МДК 08.01 Проектирование и разработка интерфейсов пользователя профессионального модуля «Разработка дизайна веб-приложений» является специальным, устанавливающим базовые знания для получения выпускником профессиональных умений, и преподается студентам специальности 09.02.07 Информационные системы и программирование.

Методические рекомендации по выполнению самостоятельной работы по МДК 08.01 Проектирование и разработка интерфейсов разработаны в соответствии с рабочей программой ПМ.08 Разработка дизайна веб-приложений.

Содержание методических рекомендаций по выполнению самостоятельной работы по данному МДК соответствует требованиям Федерального государственного образовательного стандарта по специальностям среднего профессионального образования.

По учебному плану в соответствии с рабочей программой учебной ПМ.08 Разработка дизайна веб-приложений студентами очной формы обучения предусмотрено самостоятельных занятий – 4 часа, консультаций - 12 часов.

Целью методических рекомендаций является обеспечение эффективности самостоятельной работы студентов с литературой на основе организации её изучения.

Задачами методических рекомендаций по самостоятельной работе являются:

- активизация самостоятельной работы студентов;
- содействие развития творческого отношения к данному МДК;
- выработка умений и навыков рациональной работы с литературой;
- управление познавательной деятельностью студентов.

Функциями методических рекомендаций по самостоятельной работе являются:

- определение содержания работы студентов по овладению программным материалом;
- установление требований к результатам изучения МДК.

Сроки выполнения и виды отчётности самостоятельной работы определяются преподавателем и доводятся до сведения студентов.

МДК 08.01 Проектирование и разработка интерфейсов базируется на знаниях, умениях и навыках, полученных студентами при изучении дисциплины «Информатика». Использование междисциплинарных связей обеспечивает системность изучения материала дисциплины и исключение дублирования.

В соответствии с рабочей программой ПМ.08 Разработка дизайна веб-приложений студент должен:

иметь практический опыт

- в разработке дизайна веб-приложений в соответствии со стандартами и требованиями заказчика;
- создании, использовании и оптимизировании изображений для веб-приложений;
- разработке интерфейса пользователя для веб-приложений с использованием современных стандартов;

уметь:

- создавать, использовать и оптимизировать изображения для веб-приложений;
- выбирать наиболее подходящее для целевого рынка дизайнерское решение;
- создавать дизайн с применением промежуточных эскизов, требований к эргономике и технической эстетике;
- разрабатывать интерфейс пользователя для веб-приложений с использованием современных стандартов;

знать:

- нормы и правила выбора стилистических решений;
- современные методики разработки графического интерфейса;
- требования и нормы подготовки и использования изображений в информационно-телекоммуникационной сети "Интернет" (далее - сеть Интернет);
- государственные стандарты и требования к разработке дизайна веб-приложений.

- **ТЕМАТИЧЕСКИЙ ПЛАН ВИДОВ САМОСТОЯТЕЛЬНОЙ РАБОТЫ**

Наименование и краткая характеристика	**Количество часов**	**Вид работы**
1	**2**	**3**
Проработка учебной и специальной технической литературы Подготовка к лабораторным работам, оформление лабораторных работ, отчетов и подготовка к их защите	**4**	Подготовка докладов Оформление отчета, подготовка к защите
Консультации по МДК: 1 Использование свойств CSS2 и CSS3. 2 CSS-фреймворки. 3 Шаблоны CMS. 4 SEO – оптимизация 5 Использование языка сценариев JavaScript	**12** 2 2 2 2 4	Создание сценариев, разработка мероприятий по оптимизации веб-сайта

1. ПОРЯДОК ВЫПОЛНЕНИЯ САМОСТОЯТЕЛЬНОЙ РАБОТЫ

По каждому вопросу, выносимому на самостоятельную работу студентам, приведены методические рекомендации.

1. Проработка учебной и специальной технической литературы.

Оснащение: рекомендуемая литература, методические указания по выполнению самостоятельной работы.

*Задание:*подготовка докладов.

Порядок выполнения задания

На основании основной и дополнительной специальной технической литературы, рекомендуемой к выполнению самостоятельной работы студентам необходимо рассказать о сетях передачи данных, о методах кодирования применяемых при передаче данных, о мобильных сетях первого, второго, третьего и четвертого поколений, о современном оборудовании сетей, модели взаимодействия открытых систем, технологии глобальных систем.

Подготовка к лабораторным работам, оформление лабораторных работ, отчетов и подготовка к их защите.

Оснащение: рекомендуемая литература, методические указания по выполнению самостоятельной работы, методические указания по выполнению лабораторных работ, требования к оформлению отчета.

Задание: оформление отчета, подготовка к защите

Порядок выполнения практического задания

На основании основной и дополнительной литературы, рекомендуемой к выполнению самостоятельной работы, а также на основании методических указаний по выполнению лабораторных работ, требований к оформлению отчета студентам необходимо подготовиться к лабораторной работе, оформить отчет и подготовиться к защите. В процессе подготовки студент должен усвоить теоретический материал, относящийся к лабораторной работе, изучить и ясно представить себе содержание и порядок выполнения лабораторной работы, знать ответы на приведенные в методическом руководстве контрольные вопросы.

Консультации

1 Использование свойств CSS2 и CSS3.

Выполнение заданий, представленных на веб-ресурсе
http://belarusweb.net/css/css_zadachnik/sbornik_zadach_po_css.html

1. Загрузите учебный код адаптивного каркаса страницы, при помощи комментариев изучите его и затем сверстайте самостоятельно.
2. Загрузите учебный код адаптивного каркаса страницы, при помощи комментариев изучите его и затем сверстайте самостоятельно. Сравните с первым вариантом.
3. Загрузите учебный код адаптивного каркаса страницы, при помощи комментариев изучите его и затем сверстайте самостоятельно. Сравните с предыдущими вариантами.
4. В учебном коде, который можно загрузить здесь, показано создание боковой навигации страницы для прокрутки ее вверх или вниз. Комментарии отсутствуют, но кода немного, поэтому разобраться в нем не составит особого труда. Изучите пример, а затем сверстайте его самостоятельно.

5. Используя код предыдущего примера, создайте внизу страницы кнопку 'Наверх'. Боковую навигацию уберите. Если вы не смогли выполнить задание, посмотрите решение, а затем наберите код самостоятельно.

6. В учебном коде, который можно загрузить здесь, показано создание в боковой части сайта кнопки 'Наверх', которая появляется после прокрутки страницы на указанное число, например, пикселей. Комментариев мало, но кода немного, поэтому разобраться в нем не составит особого труда. Изучите пример, а затем сверстайте его самостоятельно.

7. Создайте горизонтальное меню, показанное на рисунке. Используйте для этого список, расположив ссылки в пунктах списка. Чтобы пункты шли строкой, преобразуйте их в строчные элементы. При наведении курсора мыши на пункты меню, они должны менять цвет на черный. Загрузить решение можно здесь. Преобразуйте меню в вертикальное. Чтобы убрать номера пунктов, используйте соответствующее свойство для маркеров списка (см. справочник CSS).

8. Изучите следующий код раскрывающегося меню, после чего наберите его самостоятельно.

9. Изучите следующий код вертикального раскрывающегося многоуровневого меню, после чего наберите его самостоятельно.

10. Изучите следующий код горизонтального плавно раскрывающегося меню, после чего наберите его самостоятельно.

Верстаем учебный сайт №1

11. Для начала полистайте странички учебного сайта, загрузите и внимательно изучите его код, а затем сверстайте сайт самостоятельно (без комментариев, только код).

Верстаем учебный сайт №2

12. Для получения дополнительной практики по верстке сайтов посмотрите еще один учебный пример сайта, загрузите и при помощи комментариев изучите его код, а затем, используя готовый макет и графические заготовки, сверстайте сайт самостоятельно.

Таблицу стилей для больших разрешений постарайтесь сделать сами. Если вы работаете на ноутбуке, то для моделирования больших разрешений измените в медиазапросах разрешения с 1366px, например, на 1266px, чтобы сработала таблица стилей для больших разрешений экранов. При создании таблицы стилей опирайтесь на коэффициент увеличения размеров 1.3. А далее смотрите на результат отображения в браузере и корректируйте.

Размеры изображений можно изменить либо в графическом редакторе, либо растянуть при помощи соответствующего свойства CSS (см. пример и справочник).

2 CSS-фреймворки

Bootstrap - это фреймворк из трёх языков HTML/CSS/JS. Благодаря большому функционалу верстать сайты становится легко и быстро, ну конечно если во всем разобраться.

Появился в стенах компании Twitter и назывался «Twitter Bootstrap». Но из-за того что его захотели сделать всемирным пришлось отказаться от слова Twitter в названии. По моему личному опыту в bootstrap есть ряд плюсов:

1. *Быстрота верстки* — большое количество готовых компонентов даёт возможность не останавливаться на обыденностях.

2. *Адаптивность* — возможность настраивать размеры блоков сайта в зависимости от ширины устройства, как для компьютера так и для телефона.

3. *Популярность* — из-за которой существует большое количество статей и уроков, а также форумов. Поэтому по любому пустяку, в котором вы сомневаетесь можете найти ответ на просторах интернета или задать вопрос на форуме.

4. Bootstrap можно использовать для создания сайтов с *различными CMS* — WordPress, Joomla, Opencart.

Как установить bootstrap

Есть два способа его подключения:

- Через **скачивание файлов**.

Зайдите на официальный сайт выберите компоненты которые вам понадобятся для работы (об этом мы поговорим на следующем уроке) и в самом конце скачайте нажав на кнопку «Compile and Download». Далее распакуйте архив у себя на компьютере.

Для базовой работы понадобится лишь подключить один файл в <head> — bootstrap.min.css. Вы можете увидеть его на рисунки сверху он подсвечен оранжевым.

<link rel='stylesheet' href='/css/bootstrap.min.css' type='text/css' media='all'>

- Или через **cdn**

Но он требует подключение к интернету во время работы с фреймворком. Для подключения bootstrap.min.css добавьте эту строчку кода в <head> —

<link rel = "stylesheet" href = "https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/css/bootst

bootstrap.min.js —

<script src="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/js/bootstrap.min.js"></script

Строение фреймворка

Bootstrap насчитывает не малое количество компонентов, которые помогают нам верстать быстрее (да где-то вы это уже слышали). Поэтому давайте посмотрим, что вошло в эти компоненты.

Вы можете подробнее познакомится с этими функциями нажав на заголовок

Сеточная система

Все сайты строятся на блоках, поэтому бутстрап уделил этому отдельное внимание сделав — сетку. Она делит родительский блок (в котором находится) на 12 одинаковых по размерам частей. Также можно объединять между собой эти части, например можем дать одному блоку 3 части, второму 6 и третьему тоже 3.

Но главное ее преимущество в том, что можно менять размер блоков в зависимости от размера экрана гаджета который используется — компьютер или телефон. Так например col-md отвечает за размер экрана шире 970 px, а col-xs за ширину менее 768 px.

Оформление текста

Вам изредка придется менять размер заголовка первого уровня, или размер цитаты. Потому что все вымерено до пикселя.

Также предусмотрена возможность использовать размер h1 заголовка (и других) для обычного текста сделав вот так . По сравнению с обычным заголовком <h1></h1> его мы можем использовать сколько нам нужно раз.

2. Шаблоны CMS

WordPress в данный момент является наиболее популярной системой управления контентом (CMS) в интернете, занимая рыночную долю в 66%. Примерно 30% всех сайтов в сети работает на базе WordPress.

Типы тем

Сегодня существуют WordPress-темы под самые разные цели. Есть узкоспециализированные темы, которые позволяют создавать сайты для врачей, отелей, сайты в виде досок объявлений, сайты-портфолио и т.д. Существуют и более универсальные темы, нацеленные на блогеров, владельцев бизнеса и т.д. Большинство тем попадают в одну из следующих категорий:

Блоггинг. Темы, разработанные в основном для ведения блога

Бизнес. Темы для создания корпоративных сайтов.

Портфолио. Темы для вывода изображений и видеофайлов.

Журналы. Аналогичны блоговым темам, однако имеют более сложную разметку

Электронная коммерция. Используются для создания онлайн-магазинов

Универсальные темы. Крупные темы, которые могут использоваться для разных целей.

Приложения. Более сложные темы, разработанные под определенное решение – к примеру, темы для создания каталогов, досок объявлений, темы для агентств недвижимости.

Фреймворки. Фундамент для создания ваших собственных тем.

Разные типы тем предлагают разную стилизацию и функциональность, поэтому надо обязательно читать описание темы и смотреть демонстрационный вариант темы, чтобы понять, отвечает ли она вашим требованиям. Далее мы посмотрим на то, где можно скачать платные и бесплатные темы, в чем состоит разница между ними, а также как установить их на свой сайт.

3. SEO – оптимизация

Основные этапы работ SEO продвижения сайта

Поисковое продвижение сайта или SEO (англ. search engine optimization) – это комплекс работ над «улучшением» сайта для увеличения конверсий и поднятия ключевых запросов в рейтинге поисковых систем. То есть другими словами, задача сео-специалиста сделать сайт, отвечающий всем требованиям пользователя по данному запросу.

Поисковые системы – это умные роботы, которые давно разработали Основные этапы продвижения сайта

1 Этап - Анализ конкурентов и тематики сайта

На первом этапе необходимо понять на сколько конкурента тематика сайта, как позиционируются и подвигаются ресурсы, занимающие лидирующие позиции в топе. Такой анализ поможет предположить какие результаты можно ожидать через некоторое время и помогает составить оптимальную стратегию продвижения сайта.

2 Этап. Составление семантического ядра.

Составление семантического ядра или подбор ключевых запросов – немаловажный этап продвижения сайта, от которого зависит успешность всего проекта. На начальном этапе подбирается максимальное количество «ключевиков» подходящих по тематике сайта.

Все «ключи» группируются, выделяются наиболее перспективные и приоритетные направления.

После утверждения списка запросов с заказчиком – составляется сео план. В сео плане каждому запросу присваивается продвигаемая страница. В случае, если релевантной страницы нет - мы создаем новую страницу и оптимизируем ее под данный запрос.

3 Этап. Внутренняя оптимизация сайта

Оптимизацию сайта условно можно разделить на 2 составляющие: внутреннюю и внешнюю.

Внутренняя оптимизация включает в себя работы связанные с содержимым сайта, внутренним наполнением. Это:

Оптимизация структуры сайта, внутренняя перелинковка(продумывание структуры сайта, добавление, удаление, изменение расположения некоторых разделов и тд.)

Техническая оптимизация(поиск и удаление ошибок в коде сайта, поиск дублей страниц, проверка работы сервера, создание и размещение файлов robots.txt, sitemap.xml, увеличение скорости загрузки страниц, настройка микроразметки и тд)

Оптимизация контента(написание уникальных текстов с оптимальным количеством ключевых фраз, выделенных заголовками, абзацами, картинками, с добавлением таблиц и других необходимых элементов, составление уникальных метатегов, изменение заголовков, и тп)

Улучшение представления сайта в выдаче поисковых систем(анализируется тег Title и URL страниц, быстрые ссылки, сниппет, микроразметка, устанавливаются фавикон и ссылки на соц сети, сайт регистрируется в Яндекс Картах и Google Адресах для корректного отображения контактных данных.)

Улучшение юзабилити сайта или работа над улучшением поведенческих факторов(анализируется количество конверсий с сайта, ведется работа над уменьшением показателя отказов на сайте, увеличением времени пребывания на сайте и количества просмотренных страниц, ведется анализ над всеми неэффективными показателями, после чего составляется список доработок на сайте)

Это очень важная часть работ, так как пользователи, заходя на сайт, видят все изменения и оценивают на сколько сайт для них интересен и полезен, вызывает ли он у них доверие и желание заказать Ваш товар или услугу.

4 Этап. Внешняя оптимизация сайта

Внешняя оптимизация – это метод влияния на ранжирование сайта с «внешней стороны», то есть без изменения его внутренней составляющей. Термин достаточно обширный, но чаще всего под ним подразумевают работу над внешней ссылочной массой.

В данном случае, оценивается на сколько больше и авторитетнее ресурсов ссылаются (ставят ссылки) на наш сайт.

На этом этапе происходит создание ссылочного плана, составление анкор листа, поиск и анализ сайтов-доноров, анализ ссылочной массы конкурентов.

5 Этап. Развитие ресурса, привлечение дополнительного трафика

В данный этап входят работы по привлечению трафика с дополнительных источников – настраивается контекстная реклама, создаются группы в соцсетях, настраивается таргетированная реклама.

Продвижение и развитие сайтов это очень длительный и сложный процесс. Правильное составление стратегии продвижения гарантирует успешность проекта и хорошие показатели конверсий. Постоянное обновление алгоритмов поисковых систем заставляет изменять стратегию продвижения, искать новые способы улучшения ресурса и качества представляемой информации.

4. Использование языка сценариев JavaScript

1. Работа с классами

Дан узел DOM. Сделай функции hasClass(node, klass), addClass(node, klass), removeClass(node, klass), которые позволяют проверить, есть ли у элемента заданный CSS-класс, добавить к нему класс (если его еще нет) и удалить класс.

Учти, что у элемента может быть несколько классов, которые могут быть разделены одним или несколькими пробельными символами (пробел, \t, \f, \r, перевод строки \n — все эти символы ищутся с помощью \s в регулярке). Ты можешь спросить, что за идиот придумал разделять классы с помощью непонятных спецсимволов типа \f? Не знаю, но так написано в стандарте.

Если удалены все классы, то удалять аттрибут class="" не надо, пусть остается.

Примеры:

```
// вспомогательная функция для создания ноды
function createNode(name, klasses) {
    var n = document.createElement(name);
    n.className = klasses;
    return n;
}

function l(x) {
    console.log(x);
}

l(hasClass(createNode('div', 'test'), 'test')); // true
l(hasClass(createNode('div', 'test'), 'tes')); // false

l(hasClass(createNode('div', 'test1 test2'), 'tes')); // false
l(hasClass(createNode('div', 'test1 test2'), 'test1')); // true
```

В современных браузерах и HTML 5 у узлов DOM есть свойство classList:

- https://developer.mozilla.org/en-US/docs/Web/API/Element.classList (англ.)
- http://html5.by/blog/javascript-classlist-api/

Но решение должно работать и в браузерах без classList. Вот тебе в помощь код таких функций:

- из jQuery: https://github.com/jquery/jquery/blob/master/src/attributes/classes.js
- из блога: http://www.avoid.org/javascript-addclassremoveclass-functions/ (англ.)

2. Поле

Сделай поле из белых клеточек (клеточка может иметь размер около 28×28 пикселей). При клике на клеточку она должна менять цвет на черный. Под таблицей должна быть кнопка «поменять цвета». При ее нажатии все цвета клеточек меняются на противоположные.

Делать поле удобно с помощью элемента <table>. Саму таблицу надо не вставить в исходный код, а сгенерировать и добавить в DOM страницы яваскриптом.

У тебя может возникнуть желание поставить обработчик события на каждую клеточку. Не делай так, это неэффективно, достаточно одного обработчика на всю таблицу (так как события всплывают от элемента вверх по дереву DOM и можно ловить все события одним обработиком на таблице).

Чтобы поменять цвета всех клеточек сразу, необязательно обходить их в цикле. Если помечать нажатые клетки определенным классом, то перекрасить их все одновременно можно, поменяв класс на самой таблице.

Ты можешь заметить, что событие click срабатывает после отпускания левой кнопки мыши, а mousedown — при нажатии (любой) и с ним получается ощущение более быстрого отклика.

Ты можешь заметить, что, если быстро кликать по клеткам, браузер пытается выделять ячейки таблицы, и выглядит это некрасиво. Если это так, то надо средствами CSS3 сделать таблицу невыделяемой.

Информация по событиям: http://learn.javascript.ru/events

3. Сапер

Сделал поле из предыдущей задачи? Отлично, давай превратим его в игру «Сапер». Wiki: http://ru.wikipedia.org/wiki/%D0%A1%D0%B0%D0%BF%D1%91%D1%80_(%D0%B8%D0%B3%D1%80%D0%B0)

Идея игры такая: на игровом поле где-то спрятаны мины. Игрок кликает по клеткам, открывая их. Если в клетке была мина, игрок проиграл. Если нет, то в клетке выводится цифра, показывающая общее число мин в соседних 8 клетках. Если игрок открыл все клетки, кроме заминированных, он победил. Если игрок открывает клетку, рядом с которой нет мин, то все соседние клетки открываются автоматически (если на них тоже нет мин, то процесс продолжается).

Правой кнопкой мыши на неоткрытых клетках можно расставлять флажки.

Надпись «Вы победили» или «Вы проиграли» должна выводиться в окошке поверх игрового поля и содержать кнопку «Новая игра».

В качестве иконки для бомбы и флажка можешь взять какой-нибудь юникодный символ отсюда: http://unicode-table.com/ru/#miscellaneous-symbols

Как создать всплывающее окошко? Идея такая: делаем шаблон окошка и встраиваем его в страницу, примерно так:

```
<script type="text/x-template" id="template-popup">
    <div class="popup-body">
        {{text}}
        <button type="button">Новая игра</button>
    </div>
</script>
```

Заметь, я использовал тег script чтобы шаблон воспринимался браузером не как часть HTML-кода страницы, а как текст который не надо никак интерпретировать (браузер не будет пытаться выполнить код как яваскрипт, так как в аттрибуте type стоит не text/javascript. Буква x нужна так как мы придумали свое, нестандартное значение). Затем, когда требуется вывести окошко, берем текст этого шаблона, заменяем в нем конструкции вроде {{text}} на нужное нам значение, создаем элемент div, вставляем в него получившийся HTML и добавляем div в DOM. Для закрытия окошка — удаляем этот div.

Ну и разумеется требуется написать CSS-код чтобы окошко например выводилось по центру экрана.

Почему я предлагаю встроить шаблон в страницу, а не поместить его в переменную в JS коде, как здесь?

```
var template = '<div class="popup-body">\n\
        {{text}}....\n\
';
```

Потому что, во-первых, писать HTML внутри JS-строки очень неудобно, а во-вторых, HTML-разметка должна храниться в HTML-файле, а не в JS.

Задача снабжена подробными комментариями, она поможет освоить тебе разделение кода по MVC, отслеживание изменений в модели, реализацию отмены сделанных ходов, а полученные знания наверняка пригодятся тебе при написании более сложных приложений. Упоминаются knockout, angular, react.

4. Поиск по селектору

Напиши функцию dom.find(selector, context) которая ищет все элементы, соответствующие селектору selectorвнутри элемента context (если он не указан, то во всем документе). selector может иметь такой вид:

- для поиска всех тегов с определенным именем: div
- для поиска по классу: .some-class
- для поиска по id: #element

Функция должна быть максимально кроссбраузерной. В современных браузерах есть такие функции для поиска: querySelectorAll(), getElementsByClassName(), getElementsByTagName(), document.getElementById(). В старых браузерах первая и вторая могут отсутствовать, и возможно для них придется просто искать элемент полным обходом дерева DOM (что конечно медленнее, чем использовать готовую функцию).

Стоит учесть, что функция document.getElementById() не ищет отсоединенные элементы и ветки элементов DOM, не вставленные в документ с помощью функций вроде appendChild().

Собственно, если querySelectorAll() есть, то можно сразу передать ей селектор, так как эта функция ищет все элементы, соответствующие указанному CSS селектору.

Информация: http://learn.javascript.ru/searching-elements-dom

Решил задачу и хочешь усложнить себе жизнь? Сделай дополнительно такие возможности:

- можно указать несколько условий для элемента: div.class1.class2#id3
- можно искать элемент, находящийся внутри другого: div.class1 a.class2

Для этого я советую разобрать селектор на массив условий вида { tagName: null, classes: ['a', 'b'], id: null } с которыми работать будет проще. То есть селектор #id1 div.class1.class2 a превратится в массив:

```
[
    { tagName: null, id: 'id1', classes: [] },
    { tagName: 'div', id: null, classes: ['class1', 'class2']  },
    { tagName: 'a', id: null, classes: [] }
]
```

Распарсить можно например регулярными выражениями.

Функция поиска по селектору (причем с более мощными возможностями) есть во многих библиотеках, например, в JQuery, где она вызывается так:

```
var elements = $('div.class1 span.class2');
console.log(elements.length); // число найденных элементов
console.log(elements[0]); // Первый из найденных элементов
```

5. Всплывающее окно (сложная)

Напиши JS-код, который позволит привязать к любому элементу высплывающее окно (попап), которое будет появляться по клику (а для полей ввода — при установке курсора в поле). Повторный клик убирает окно (для поля ввода надо убрать курсор из поля).

Естественно, мы хотим, чтобы добавить такое окно на страницу можно было как можно проще. Для этого я предлагаю сделать такую разметку. На кнопке или поле ввода мы указываем текст пояснения и сторону, с которой оно должно появиться (left|right|above|below, по умолчанию below):

```
<input type="email"
    data-popup="Этот адрес будет использоваться...."
    data-popup-side="right">
```

Для окошка, содержащего сложную разметку, мы помещаем HTML-разметку в блок script, а в аттрибуте указщываем id этого скрипта:

```
<button type="button"
    data-popup-id="template-login-popup"
    data-popup-side="below">войти на сайт</button>

<!-- HTML-разметка для окошка -->
<script type="text/x-template" id="template-login-popup">
    <div>.....введите ваш логин и пароль ....
</script>
```

Размер всплывающего окошка должен определяться содержимым. Вставлять DOM-элемент окошка лучше всего перед кнопкой. Тебе наверно понадобятся CSSOM-свойсва и функции: offsetWidth, offsetHeight, clientWidth, clientHeight, getBoundingClientRect(), scrollLeft, scrollTop, offsetLeft, offsetTop. Надо понимать, что кнопка может быть спозиционирована по-разному, и ее положение например может меняться при изменении размера окна браузера.

Ловить события клика по кнопкам можно одним обработчиком на документе. Событие focus не всплывает и ловить события фокуса поля ввода на документе придется по-хитрому (в IE — событие focusin, в новых браузерах — захватом события focus, это описано по ссылке ниже).

Информация: http://learn.javascript.ru/focus-blur

Усложненный вариант (правда сложный): улучши определение расположения окошка в зависимости от положения элемента в окне браузера.

Если места с одной стороны от кнопки недостаточно, окошко может появиться с противоположной стороны. Окошко выравнивается по центру с элементом, но, если места недостаточно, может сдвигаться. То, есть можно использовать такой алгоритм. Допустим, мы хотим показать окошко справа от кнопки.

- определяем сколько свободного места есть справа и сколько слева
- создаем див слева и сверху от страницы (чтобы он был не виден), пробуем поместить в него контент и проверяем, какую ширину он занимает. Таким образом мы определим ширину и высоту содержимого окошка.
- если оно поместится справа, то размещаем окошко справа. Иначе, если слева места больше и оно туда поместится, то слева

- выравниваем середину окошка с серединой кнопки. Если при этом верхний край окошка уходит за окно браузера, и снизу есть место, сдвигаем окошко вниз. Аналогично в противоположном случае.
- если окошко уходит за верх или левый край документа, то выдвигаем его оттуда, так как в этом случае его никак не увидеть

Перечень рекомендуемых учебных изданий, Интернет-ресурсов, дополнительной литературы

1. Печатные издания

1. Немцова, Т.И., Назарова, Ю.В. Практикум по информатике: учеб. пособие/ Под редакцией Л.Г. Гагариной. Ч. I и II. – М. : Форум, 2014. – 288 с.: ил.
2. Дунаев В. В. HTML, скрипты и стили. Спб.: БХВ – Петербург, 2011 – 816 с.
3. Мэтью, Дэвид HTML5. Разработка веб-приложений / Дэвид Мэтью. - М.: Рид Групп, 2015. - 320 с.
4. Хоган, Б. HTML5 и CSS3. Веб-разработка по стандартам нового поколения / Б. Хоган. - М.: Питер, 2017. - 783 с.
5. Томлинсон, Тодд CMS Drupal 7. Руководство по разработке системы управления веб-сайтом / Тодд Томлинсон. - М.: Вильямс, 2017. - 560 с.
6. Дакетт, Джон HTML и CSS. Разработка и дизайн веб-сайтов (+ CD-ROM) / Джон Дакетт. - М.: Эксмо, 2017. - 480 с.
7. Прохоренок, Н.А. HTML, JavaScript, PHP и MySQL. Джентльменский набор Web-мастера / Н.А. Прохоренок. – СПб. : БХВ-Петербург, 2014. – 640 с.: ил. (+CD)

2. Электронные издания (электронные ресурсы)

1. Система федеральных образовательных порталов Информационно - коммуникационные техно-логии в образовании. [Электронный ресурс] – режим доступа: http://www.ict.edu.ru (2003-2017)
2. Ежемесячный электронный журнал «ПРОграммист». http://procoder.info/
3. http://ru.wikipedia.org
4. http://www.javaportal.ru
5. http://moolkin.ru/joomla/cms/staticheskie-i-dinamicheskie-web-sayty-v-chyom-raznitsa/
6. http://htmlbook.ru
7. http://helpx.adobe.com/ru/dreamweaver/using/creating-dreamweaver-template.html

3. Дополнительные источники

1. Дунаев, В. Самоучитель JavaScript / В. Дунаев. 2-е изд. - СПб. : Питер, 2013. – 400с.
2. Кирсанов, Д. Веб-дизайн: книга Дмитрия Кирсанова / Д. Кирсанов. – СПб : Символ-Плюс, 2013. – 376с.: ил.
3. Храмцов, П.Б. Основы Web-технологий: учебное пособие / П.Б. Храмцов, С.А. Брик, А.М. Русак, А.И. Сурин – 2-е изд., испр. –М. : Интернет-Университет Информационных технологий; БИНОМ. Лаборатория знаний, 2014. – 512с.
4. Яцюк, О. Основы компьютерного дизайна на базе компьютерных технологий / О.Яцюк. – СПб. : БХВ-Петербург, 2015. – 240с.: ил.

Критерии оценки самостоятельной работы:

Оценка «5» - тема раскрыта в достаточной мере, отражены ключевые определения по теме, сделаны выводы, оформление соответствует требованиям, недочетов нет.

Оценка «4» - тема раскрыта в достаточной мере, отражены не все ключевые определения по теме, сделаны выводы, есть небольшие недочеты в оформлении.

Оценка «3» - тема раскрыта не в полной мере, отражены не все ключевые определения по теме, выводы недостаточно глубокие, есть недочеты в оформлении.

Оценка «2» - тема раскрыта не в полной мере, не отражены ключевые определения по теме, выводы не сделаны, есть ошибки в оформлении