**Задачи курса «*Web*-программирование»**

|  |  |
| --- | --- |
| **Задание 4 (циклы, наблюдатели, витки)** | Разработать локальные одностраничные *web*-приложение (*LSPWA*) под управлением фреймворка *Vue.js* на языке *JavaScript* в соответствии с указаниями вариантов индивидуального задания.  *Базовая структура отчёта по работе:*  1. Цель работы.  2. Формулировки задач.  2.1. Задача на витки жизненного цикла.  2.2. Задача на циклы.  2.3. Задача с наблюдателями.  3. Спецификация оборудования, на котором выполнялась работа.  4. Информация об используемом браузере.  5. Технология локального подключения фреймворка *Vue.js.*  6. Русифицированные схемы жизненного цикла *Vue.js*.  7. Таблица соответствия переменных и методов, используемых в *web*-приложении.  8. Содержательная часть  8.1. Витки жизненного цикла (код *web*-приложения + отображение в браузере + Сеть Петри).  8.2. Циклическая отрисовка (код *web*-приложения + отображение в браузере + Сеть Петри).  8.3. Наблюдатели (код *web*-приложения + отображение в браузере + Сеть Петри).  9. Вывод.  **Реализовать:**  – локально (*LSPWA*), не прибегая к инструментарию *Node.js* и *npm* (*Node Package Manager*);  – *v-for* в файле с именем *index-v-for\_Familiya\_I\_O.html;*  – *watches* в файле с именем *index-watches\_Familiya\_I\_O.html;*  – витки жизненного цикла в файле с именем *index-hooks\_Familiya\_I\_O.html.* |

|  |  |
| --- | --- |
| **Витки жизненного цикла**  (общая формулировка для творческого задания)  **Не использовать:** *v-for,**jQuerry*, «чистый» *JavaScript* | Продумать схему тестирования и демонстрации работы всех витков / хуков / методов жизненного цикла фреймворка *Vue.js* в формате одностраничного *web*-приложения, отличающегося от рассмотренного в лекционном материале курса «*Web*-программирование». Реализовать схему тестирования строго под *Vue.js 3.x* (**Приложение 1**). Русифицировать схему жизненного цикла *Vue.js 3.x* (допускается использование готовой реализации из официальной документации).  Близкая реализация *web*-приложения для тестирования под *Vue.js 2.x* приведена в **Приложении 2**.  Привести информацию об используемом браузере по аналогии с иллюстрациями, представленными в **Приложении 3**.  Составить таблицу соответствия переменных и методов, используемых в *web*-приложении. Структура таблицы соответствия представлена в **Приложении 4**.  Заполнить спецификацию оборудования, используемого для выполнения работы **Приложение 5**.  **Не использовать:** *jQuerry*, «чистый» *JavaScript*, директиву *v-for* (Приложение 2).  **Реализовать:** локально, не прибегая к инструментарию *Node.js* и *npm* (*Node Package Manager*) – раскрыть данный вопрос в пункте 5 отчёта.  *Рекомендации:* для отработки отдельных витков / хуков / методов жизненного цикла неизбежно потребуется воспользоваться встроенной браузерной консолью. |

|  |  |
| --- | --- |
| **Списки, перебор**  ***v-for***  (**Не использовать:***jQuerry*, «чистый» *JavaScript*) | **Вариант 1.** В области для размещения масштабируемой векторной графики <*svg*> размером 400 на 400 пикселей по всему контуру, отстоящему на 100 пикселей внутрь, разместить окружности радиуса 2 с шагом 5 пикселей. Предоставить оператору возможность для изменения радиусов сразу всех окружностей посредством элемента <*input*> в режиме *range* с единичным шагом. Диапазон изменения радиуса автор определяет самостоятельно, но руководствуясь рациональными и аргументированными соображениями (записать их в отчёте текстом).  **Вариант 2.** Сделать статический, нумерованный список (гипертекстовая разметка – <*ol*>) по тематике «Марки автомобилей». Предоставить возможность оператору для добавления в список элементов – для ввода наименований автомобилей использовать окно *prompt*. Предоставить возможность оператору для удаления из списка элементов. Если в списке остаётся только один элемент – запретить его удаление. Для удаления / добавления использовать экранные кнопки <*button*>.  **Вариант 3.** В обрамлённую таблицу <*table*>, состоящую из одного столбца выводится список зарегистрированных пользователей в формате [Фамилия];[Имя];[Отчество];[Телефон]. Под таблицей размещены поля для ввода <*input*> соответствующих значений фамилии, имени, отчества и телефона пользователя. Нажатие на «*Enter*» в любом из полей – сигнал для ввода данных. При этом каждое из полей контролируется на предмет отсутствия пустого значения. Если хотя бы одно из полей остаётся пустым – запрещать ввод и демонстрировать уведомление в *alert*. Удачный ввод сопровождается отображением нового пользователя в таблице. Содержимое полей <*input*> после удачного ввода сбрасывается.  **Вариант 4.** Дан кубический массив размерности, указываемой пользователем в <*input*>. Массив заполняется и перезаполняется псевдослучайным образом каждый раз, как только меняется значение в <*input*>, но только после потери этим элементом фокуса. Все генерируемые значения лежат в диапазоне не более, чем с трёхразрядными целыми десятичными значениями. Продумать способ вывода этих значений в обрамлённую таблицу по спирали. Под таблицей для контроля её заполнения выводить те же значения друг за другом в абзац текста <*p*><*/p*> в порядке увеличения индексов в традиционной последовательности чтения книги: столбец, строка, слой. Через разделитель «$».  **Вариант 5.** Некоторое изображение нарезано стык-в-стык на 9 частей таким образом, чтобы оно фрагментарно размещалось в необрамлённой таблице <*table*> размерности [*3 x 3*] при использовании директивы *v-for*.  **Вариант 6.** Изобразить в области масштабируемой векторной графики <*svg*> размерности 960 на 540 пикселей по ширине и высоте, соответственно, горизонтальную координатную ось со стрелкой и засечками. Ось отстаёт от рамок <*svg*> на 20 пикселей, соответственно. Стрелка оси всегда фиксированных размеров, устанавливаемых разработчиком (оператор не управляет настройками вида этого элемента). Оператор меняет шаг отрисовки засечек, а также их высоту посредством элементов <*input*> в режиме *range* в разумных пределах (аргументировать выбор текстом в отчёте). В качестве элементов внутри <*svg*> использовать только <*line*>. |

|  |  |
| --- | --- |
| **Списки, перебор**  ***v-for*** | **Вариант 7.**Сделать статический, маркированный, двухуровневый список (гипертекстовая разметка – <*ul*>) с маркерами различного типа на разных уровнях по тематике «Транспорт». Верхний уровень – «Категория», например, водный, железнодорожный и так далее; нижний уровень «Разновидность, относящаяся к категории», например, «лодка», «катер», «дрезина», «поезд пригородного сообщения» и т.д. Предоставить оператору возможность добавления в список элементов на каждый уровень списка при помощи окна *prompt*. Способ разграничения добавления элементов на уровни разработчику необходимо продумать самостоятельно.  **Вариант 8.** Дан массив размерности [*3 x 3*], содержащий целые числа. Выводить его содержимое в нумерованный список (<*ol*>). Предоставлять возможность оператору для добавления в список элементов, причём сразу строкой из трёх значений, разделённых запятыми в <*input*>. Подтверждать ввод по «*Ctrl*» + «*Enter*». Предоставить оператору возможность для удаления из массива значений построчно по нажатии «*Shift*» + «*Enter*» при наличии фокуса в <*input*>.  **Вариант 9.**В обрамлённую таблицу <*table*>, состоящую из четырёх столбцов, выводится список товаров в формате [Наименование] [Производитель] [Количество] [Цена]. Под таблицей размещены поля для ввода <*input*> соответствующих значений наименования товара, производителя, количества и цены. Нажатие на «*Enter*» в любом из полей – сигнал для ввода данных. При этом каждое из полей контролируется на предмет отсутствия пустого значения. Для наименования и производителя пустоту менять на серию из пяти подряд идущих дефисов «-». Для количества и цены пустоту менять на нули. Ввод сопровождается отображением нового товара в таблице. Содержимое полей <*input*> после ввода сбрасывается.  **Вариант 10.** Некоторое изображение нарезано стык в стык на 6 частей таким образом, чтобы оно фрагментарно размещалось в обрамлённой таблице <*table*> размерности [*2 x 3*] при использовании директивы *v-for*. При указании оператором в <*input*> меньшей и сопоставимой размерности таблицы не производить изменений при выводе фрагментированного изображения. При указании большей размерности – стараться настолько, насколько это возможно, центрировать фрагментированное изображение в таблице.  **Вариант 11.**Константно задан трёхмерный массив размерности [*3 x 3 x 3*], содержащий строковые значения (слова), иметь возможность «прокручивать» этот массив в таблице <*table*> размерности, задаваемой оператором через <*input*> (не обязательно квадратной размерности). Под «прокруткой» понимать изменение посредством экранных кнопок <*button*> «Вперёд» / «Назад» величины сдвига по индексам массива относительно начального элемента с комбинацией индексов вида: «[0, 0, 0]». |

|  |  |
| --- | --- |
| **Списки, перебор**  ***v-for*** | **Вариант 12.**Изобразить в области масштабируемой векторной графики <*svg*> размерности 1000 на 300 пикселей по ширине и высоте, соответственно, горизонтальную координатную ось со стрелкой, засечками, числовыми значениями под засечками и особой, более жирной засечкой в ноле (присутствует всегда, когда указан диапазон с различными знаками (*sign*) значений (*values*) в начале и конце, даже в ситуациях, когда шаг отрисовки засечек в точности не попадает в ноль), разграничивающей области отрицательных и положительных значений. Ось отстаёт от рамок области <*svg*> на 100 пикселей как сверху, так и по бокам. Стрелка оси всегда фиксированных размеров, устанавливаемых разработчиком (оператор не управляет настройками вида этого элемента). Оператор меняет шаг отрисовки засечек, указывает начало и конец диапазона, указывает значение округления для текста маркеров под засечками при использовании элементов <*input*>. В качестве основных элементов внутри <*svg*> использовать <*line*>, для маркеров, соответственно, использовать <*text*>.  **Вариант 13.**Изобразить внутри обрамлённой области масштабируемой векторной графики <*svg*> с фиксированными значениями ширины и высоты функцию вида *y(x) = sin(x)* окружностями <*circle*> настраиваемого радиуса (2 – 10 пикселей с единичным шагом в <*input*> в режиме *range*) на белом фоне без осей. Предоставить оператору возможность масштабирования с единичным шагом функции в фиксированном диапазоне 10%–100% через <*input*> в режиме *range*.  **Вариант 14.**Изобразить в области масштабируемой векторной графики <*svg*> размерности 540 на 960 пикселей по ширине и высоте, соответственно, вертикальную координатную ось со стрелкой и засечками. Ось отстаёт от рамок <*svg*> на 50 пикселей по вертикали и горизонтали, соответственно. Стрелка оси всегда фиксированных размеров, устанавливаемых разработчиков (оператор не управляет настройками вида этого элемента). Оператор меняет шаг засечек при использовании элементов <*input*> в режиме *range* в разумных пределах (аргументировать выбор текстом в отчёте). Между засечками посередине всегда размещаются промежуточные засечки точно посередине. Оператор указывает начало и конец диапазона вывода числовых маркеров под засечками при использовании элементов <*input*>; при использовании <*input*> в режиме *checkbox* оператор меняет направление стрелки оси: либо положительное направление, либо снизу-вверх, либо сверху-вниз. В качестве элементов внутри <*svg*> использовать только <*line*> и <*text*>.  **Вариант 15.**Заполнять обрамлённую область масштабируемой векторной графики <*svg*> с отступом в 10 пикселей целыми, десятичными, одноразрядными значениями при использовании псевдослучайной генерации *JavaScript* по размеру эквивалентными тексту в 14 пунктов в текстовом редакторе *Microsoft Office Word*. Предоставить оператору возможность изменения размеров области масштабируемой векторной графики по ширине и высоте при использовании <*input*>.  **Вариант 16.** Дан массив размерности [*4 x 7*], содержащий наименования изображений, планируемых к подстановке в ячейки таблицы. Все изображения подобрать одного размера, но различные по содержанию. Добиться эффекта, чтобы зазор между изображениями оказался минимален. Границы таблицы сделать невидимыми. |
| **Списки, перебор**  ***v-for*** | **Вариант 17.**В области масштабируемой векторной графики <*svg*> фиксированных размеров 640 на 480 пикселей по ширине и высоте, соответственно, строить отрезками с единичным шагом функцию *y(x) = x \* x*. Предоставить оператору возможность вводить параметры диапазонов рассмотрения графика функции как по абсциссе, так и по ординате, намечать засечки на осях с численными значениями под ними. Предусмотреть возможность отключения осей по отдельности, отключения подписей с числовыми значениями, отключения засечек. Для каждого состояния видимости / невидимости ввести элемент <*input*> в режиме *checkbox*.  **Вариант 18.**Некоторое изображение нарезано стык в стык на 15 частей таким образом, чтобы оно фрагментарно размещалось в обрамлённой таблице <*table*> размерности [*5 x 3*] при помощи директивы *v-for*. При указании оператором меньшей размерности не выводить ни изображение, ни таблицу на страницу. При указании оператором сопоставимой и большей размерности через <*input*> размещать фрагментированное изображение справа снизу в таблице.  **Вариант 19.**Иметь в наличии клетчатую область масштабируемой векторной графики <*svg*>. По умолчанию клетка выполнена с единичным шагом. Предоставить возможность оператору масштабировать клетчатую область. Предоставить возможность оператору для изменения размеров <*svg*> по ширине и по высоте.  **Вариант 20.**Сделать статический, маркированный, трёхуровневый список (гипертекстовая разметка – <*ul*>) с маркерами различного типа на разных уровнях по тематике «Территориально-административное деление города Москвы». Верхний уровень – «Административный округ», например, «САО», «ЗАО», «ЮВАО» и так далее; средний уровень «Район», например, «Аэропорт», «Ярославский», «Раменки», и так далее. Нижний уровень – проспекты / бульвары / улицы / проезды / переулки / тупики и так далее. Предоставить оператору возможность для добавления в список элементов на каждый уровень списка при помощи окна *prompt*. Способ разграничения добавления элементов на уровни разработчику необходимо продумать самостоятельно.  **Вариант 21.** Изобразить внутри обрамлённой области масштабируемой векторной графики <*svg*> с фиксированными значениями ширины и высоты функцию вида *y(x) = cos(x)* отрезками на белом фоне без осей. Предоставить оператору возможность масштабировать функцию в фиксированном диапазоне 100-200% с шагом 10% при использовании <*input*> в режиме *range*.  **Вариант 22.** Дана обрамлённая таблица <*table*> в каждой ячейке которой располагается <*input*> в режиме *number*. Исходно это пустой двумерный массив указанного оператором размера по строкам и столбцам в <*input*> в режиме *number*. Предусмотреть экранную кнопку для вывода заполненного массива. Пока массив полностью не сформирован (не заполнен значениями) – по нажатии на экранную кнопку выводится *alert* с побуждающим сообщением для оператора. Как только все желаемые значения введены в массив – нажатие на экранную кнопку приводит к демонстрации другой обрамлённой таблицы, в которой все ранее введённые значения – статичны (не редактируемы). |

|  |  |
| --- | --- |
| **Списки, перебор**  ***v-for*** | **Вариант 23.**Некоторое изображение нарезано стык в стык на 12 частей таким образом, чтобы оно фрагментарно размещалось в обрамлённой таблице <*table*> размерности [*6 x 2*] при помощи директивы *v-for*. При указании оператором меньшей размерности выдавать сообщение *alert* и возвращать значения по умолчанию. При указании оператором сопоставимой и большей размерности при использовании <*input*> размещать фрагментированное изображение слева снизу в таблице.  **Вариант 24.** В области масштабируемой векторной графики <*svg*> фиксированных размеров 200 на 600 пикселей по ширине и высоте, соответственно, строить окружностями радиуса 2 с единичным шагом функцию *y(x) = x*. Предоставить оператору возможность вводить параметры диапазонов рассмотрения как по абсциссе, так и по ординате, намечать засечки на осях с численными значениями под ними. Предусмотреть возможность отключения оператором осей по отдельности, отключения подписей с числовыми значениями, отключения засечек. Для каждого состояния включения / отключения ввести парные элементы <*input*> в режиме *radio*.  **Вариант 25.**Разметить обрамлённую табличную <*table*> область указанной оператором в <*input*> размерности. Каждая ячейка таблицы должна содержать в себе свои и индексы в формате «*(i, j):* ». Только после этой конструкции указывается соответствующее значение, содержащееся в массиве. Значения в массиве генерируются псевдослучайным образом после того, как один из элементов <*input*>, задающих размерность, теряет фокус. |

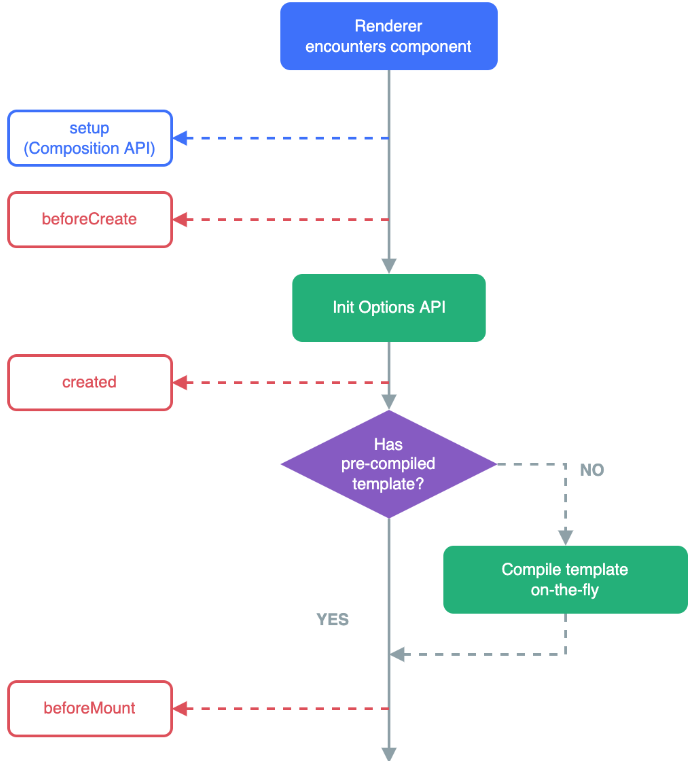
|  |  |
| --- | --- |
| **Наблюдатели**  ***watch***  ***(Не использовать:*** *jQuerry*, «чистый» *JavaScript****)*** | **Вариант 1.** Организовать транслитерацию с кириллицы на латиницу в <*input*>. Ни при каких обстоятельствах не показывать символы кириллицы – сразу переводить их в латиницу внутри <*input*>. При этом текст на латинице вводить запрещено. Об этом должны уведомлять стандартные окна *confirm*. Если пользователь с этим сообщением соглашается – продолжение ввода возможно, если нет – реализовать сброс всего накопленного текста в <*input*> и запретить дальнейший ввод.  **Вариант 2.** Размеры обрамлённого элемента для размещения масштабируемой векторной графики <*svg*> настраиваются двумя элементами <*input*> в режиме *range* с единичным шагом в диапазонах 0…640 и 0…480 по ширине и высоте, соответственно. Организовать проверку и запретить превышение совокупных размеров <*svg*> над, в точности, 0.3 мегапикселями.  **Вариант 3.** Организовать <*input*> со свойствами калькулятора. Вводить в <*input*> можно что угодно, но как только там появляется конструкции вида: «a + b», «a – b», «a \* b» или «a / b», – выводить в этот же <*input*> результат выполнения арифметической операции. В «a» и «b» подразумеваются только десятичные, одноразрядные целые значения.  **Вариант 4.** Подобрать шесть идентичных по ширине и высоте иллюстраций, выводимых в <*img*> фиксированного размера. Седьмое изображение тех же размеров содержит надпись «изображение отсутствует». Вводить в <*input*> название (без расширения) иллюстрации. Если оно соответствует одному из шести имеющихся наименований – выводить соответствующую иллюстрацию, а если нет – выводить иллюстрацию с надписью «изображение отсутствует».  **Вариант 5.**Список <*select*> наполняется последовательно пунктами, наименование которых вводятся в <*input*> и подтверждается нажатием на «*Enter*». Выставляется ограничение на ввод – запрещено вводить более 10 символов. Под это условие подобрать соответствующую, фиксированную ширину <*select*>. Учесть при подборе ширины <*select*>, что могут вводиться и все заглавные символы.  **Вариант 6.** В начале страницы в абзаце текста (<*p*><*/p*>) выводится последний заложенный (на этапе загрузки) или последний введённый (на этапе работы) строковый элемент массива. Элементы в массив вводятся через элемент <*input*> по нажатии сочетания клавиш «*Ctrl*» + «пробел». Как только в элементе ввода появляется значение-команда «@половина» или «@полный» на страницу, в абзацы текста выводится, соответственно, либо последняя половина имеющихся в массиве значений, либо полный перечень хранимых в массиве значений.  **Вариант 7.** Посредством <*input*> последовательно заполняется статический, нумерованный список по нажатии на «пробел» («*Spacebar*»). Позиции в списке – семантически значимые. Имеется ограничение на ввод пяти символов. Как только в <*input*> вводится значение, превышающее 5 символов – содержимое нумерованного списка принудительно опустошается без возможности восстановления. |

|  |  |
| --- | --- |
| **Наблюдатели**  ***watch*** | **Вариант 8.** Работа программы начинается с вывода окна *prompt*, призывающего ввести количество попыток. Далее пресекать попытки ввода в <*input*> пользователем числовых значений. На каждое числовое значение выдавать *alert* с предупреждением, что ввод будет заблокирован через *N* попыток. После *N* попыток ввести в <*input*> числовые значения блокировать <*input*> на ввод чего-либо, но не средствами атрибута *disabled*, а теми же средствами *watch*, встроенными во *Vue.js*.  **Вариант 9.**На страницу выводится обрамлённый элемент масштабируемой векторной графики с значениями ширины и высоты в пропорции 16:9. Начальные приближения ширины и высоты записаны в переменные, значения которых выводятся в соответствующие элементы <*input*> Синхронно подгонять под пропорции 16:9 любые изменения, связанные как с шириной, так и с высотой.  **Вариант 10.** Организовать <*input*> со свойствами калькулятора. Вводить в <*input*> можно что угодно, но реакция на изменение содержимого будет выполняться лишь в случаях соответствия текста одному из шаблонов: «*a+b*», «*a–b*», «*a\*b*» или «*a/b*», – выводить в этот же <*input*> результат выполнения арифметической операции. В «a» и «b» могут быть любые вещественные значения. Десятичный разделитель – «,». Предусмотреть настройки, когда количество пробелов в начале и в конце шаблона не имеет абсолютно никакого значения.  **Вариант 11.** Создать конвертер градусы / радианы / грады. Любая вводимая величина влияет на остальные. Отдельных кнопок конвертации не предусмотрено. Оформить конвертер таблично.  **Вариант 12.**Заполняется статический маркированный список <*ul*>. Ввод новых значений предусмотрен через <*input*> по нажатии сочетания «*Ctrl*» + «*Enter*». Как только в поле <*input*> возникает значение-команда «#редактировать#» статический список меняется на функциональный <*select*>, в котором можно выбирать желаемую для редактирования позицию, её текст передаётся в <input>, где можно его отредактировать и по «*Ctrl*» + «*Enter*» изменить в перечне. Значение-команда «#вводить#» переводит список обратно в состояние статического, маркированного.  **Вариант 13.** Создать конвертер валют. Любая вводимая величина влияет на остальные. Отдельных кнопок конвертации не предусмотрено. В процессе конвертации участвуют рубли, доллары, евро, йены, юани, рупии. Взять к рассмотрению курс, зафиксированный на момент: 10 января 2024 года. Оформить конвертер таблично.  **Вариант 14.** Организовать <*input*> со свойствами калькулятора. Вводить в <*input*> можно что угодно, но как только там, среди текста, появляется конструкции вида: «*Sin(x)*», «*Cos(x)*», «*Tg(x)*», «*Ctg(x)*», «*pi*» – выводить в этот же <*input*> на соответствующих позициях расчётные значения тригонометрических функций и константы.  **Вариант 15.**Пресекать попытки ввода в <*input*> пользователем кириллицы. Ни при каких обстоятельствах даже не демонстрировать кириллицу в <*input*>. На каждый кириллический символ выдавать *alert* с предупреждением, что ввод будет заблокирован. После трёх попыток ввести в <*input*> кириллицу. Блокировать <*input*> на ввод чего-либо, но не средствами атрибута *disabled*, а теми же средствами *watch*, встроенными во *Vue.js*. |

|  |  |
| --- | --- |
| **Наблюдатели**  ***watch*** | **Вариант 16.** Создать конвертер температур. Любая вводимая величина влияет на остальные. Отдельных кнопок конвертации не предусмотрено. В процессе конвертации участвуют: цельсии, кельвины, рёмеры, реомюры, ньютоны, делили и фаренгейты. Оформить конвертер таблично.  **Вариант 17.** Список <*select*> наполняется последовательно пунктами, наименование которых вводятся в <*input*> и подтверждается нажатием на «пробел» («*Spacebar*»). Выставляется ограничение на ввод – запрещено вводить более 15 символов. Как только очередной вводимый элемент составляет более 15 символов – содержимое списка очищается. Предусмотреть возможность возврата значений списка по «*Ctrl*» + «*z*» при фокусе на <*input*>.  **Вариант 18.** Подобрать четыре идентичных по ширине и высоте иллюстрации, выводимые в <*img*> фиксированного размера. Пятое изображение тех же размеров содержит надпись «изображение отсутствует». Частично вводить в <*input*> название (без расширения) иллюстрации. По нажатии на «*Ctrl*» + «*Enter*» пытаться подобрать подходящее из известного перечня. Если таковое находится среди имеющихся наименований – выводить соответствующую иллюстрацию и дополнять текст недостающим фрагментом, а если нет – выводить иллюстрацию с надписью «изображение отсутствует» и сбрасывать ранее введённый фрагмент.  **Вариант 19.**Создать конвертер единиц информации. Любая вводимая величина влияет на остальные. Отдельных кнопок конвертации не предусмотрено. В процессе конвертации участвуют: биты, байты, килобайты, мегабайты, гигабайты, терабайты, петабайты, экзабайты, зеттабайты, йоттабайты. Оформить конвертер таблично.  **Вариант 20.** Размеры обрамлённого элемента для размещения масштабируемой векторной графики <*svg*> настраиваются двумя элементами <*input*> в режиме *number* с единичным шагом в диапазонах 0…1280 и 0…720 по ширине и высоте, соответственно. Организовать проверку и запретить превышение совокупных размеров <*svg*> над значением, которое считается 1-м мегапикселем (0.9 мегапикселя).  **Вариант 21.** На страницу выводится обрамлённый элемент масштабируемой векторной графики с значениями ширины и высоты в пропорции 3:4. Начальные приближения ширины и высоты записаны в переменные, значения которых выводятся в соответствующие элементы <*input*> Синхронно подгонять под пропорции 3:4 любые изменения, связанные как с шириной, так и с высотой. Выводить попарно только целые значения. При получении дробных значений не округлять результат, а подводить оба значения ширины и высоты под ближайшие целые значения.  **Вариант 22.**Создать конвертер по «традиционным» системам счисления. Любая вводимая величина влияет на остальные. Отдельных кнопок конвертации не предусмотрено. В процессе конвертации участвуют: двоичная, восьмеричная, десятичная и шестнадцатеричная системы. Оформить конвертер таблично.  **Вариант 23.** Организовать <*input*> со свойствами калькулятора. Вводить в <*input*> можно что угодно, но, когда там появляется только одна из конструкций вида: «*Ln(x)*», «*Lg(x)*», «*Log(x,y)*», «*Exp(x)*», «*exponent*» – выводить в этот же <*input*> расчётные значения логарифмических функций и константы. |

|  |  |
| --- | --- |
| **Наблюдатели**  ***watch*** | **Вариант 24.** В начале страницы в абзаце текста (<*p*><*/p*>) выводится последний заложенный (на этапе загрузки) или последний введённый (на этапе работы) строковый элемент массива. Элементы в массив вводятся через элемент <*input*> по нажатии сочетания клавиш «*Ctrl*» + «*p*». Как только в элементе ввода появляется значение-команда «$четверть$», «!четверть!», «%четверть%» «&четверть&» на страницу, в абзацы текста выводится, соответственно, либо первая, вторая, третья или четвёртая четверть имеющихся в массиве значений. Полный перечень хранимых в массиве значений выводится при появлении значения-команды «$!%&».  **Вариант 25.**Размеры обрамлённого элемента для размещения масштабируемой векторной графики <*svg*> настраиваются двумя элементами <*input*> в режиме *text* с единичным шагом в диапазонах 0…1600 и 0…1200 по ширине и высоте, соответственно. Организовать проверку и запретить превышение совокупных размеров <*svg*> над значением, которое считается 2-мя мегапикселями (1,9 мегапикселей). По превышении желаемого размера возвращать предыдущее зафиксированное значение. |

**Приложение 1. Схема жизненного цикла (на английском языке) одностраничного *web*-приложения под управлением *Vue.js 3.x.***



**1**

Рисунок 1 – Витки жизненного цикла экземпляра *Vue 3.x* (начало)

**1**

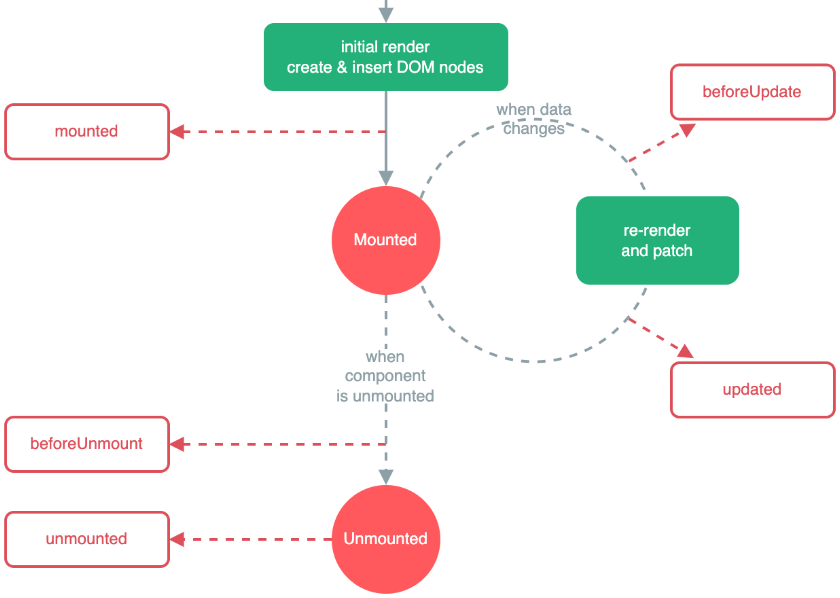
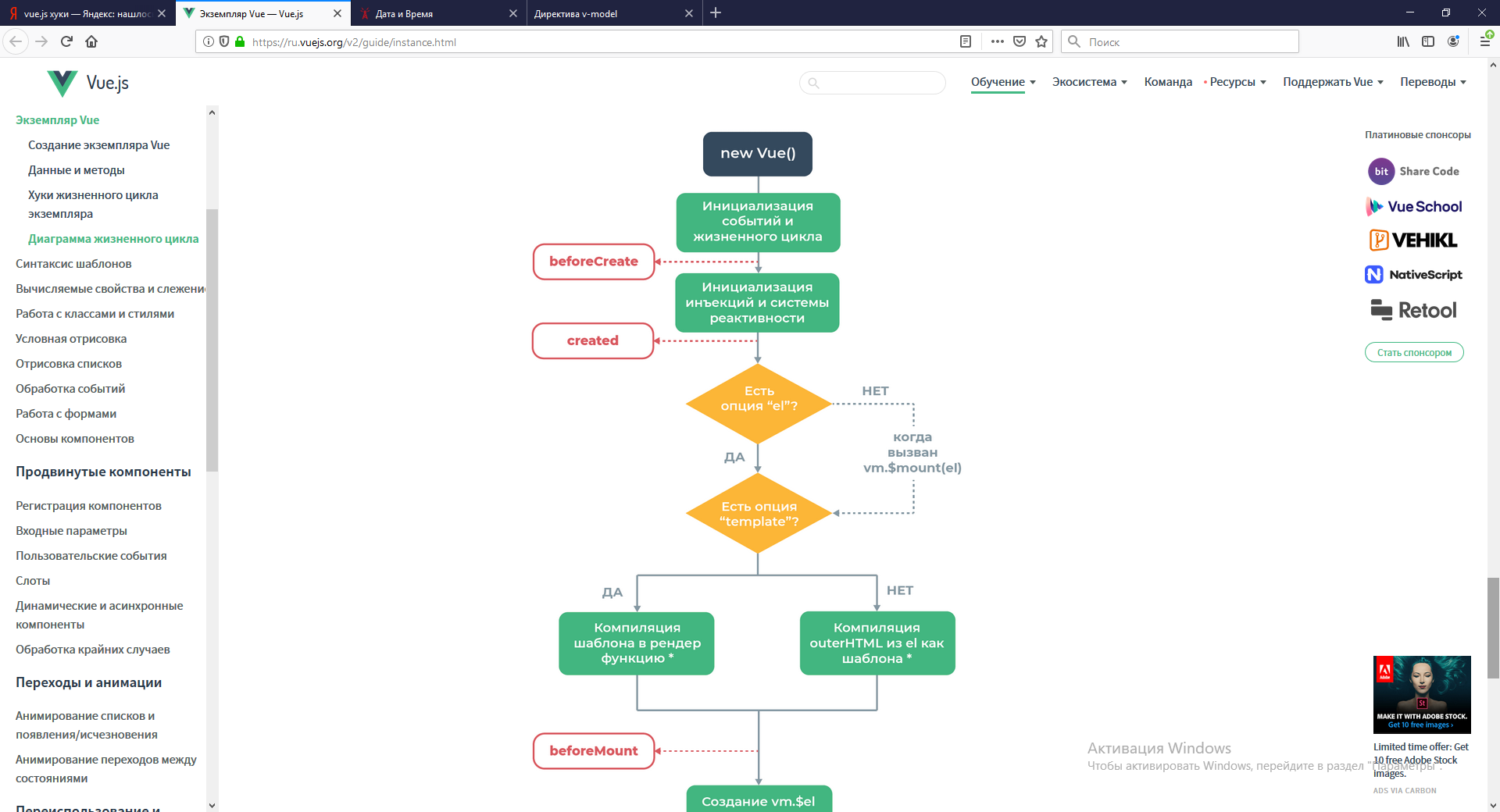


Рисунок 1 – Витки жизненного цикла экземпляра *Vue 3.x* (продолжение)

**Приложение 2. Схема жизненного цикла (на русском языке) одностраничного *web*-приложения под управлением *Vue.js 2.x.***



**1**

Рисунок 2 – Витки жизненного цикла экземпляра *Vue 2.x* (начало)

**1**

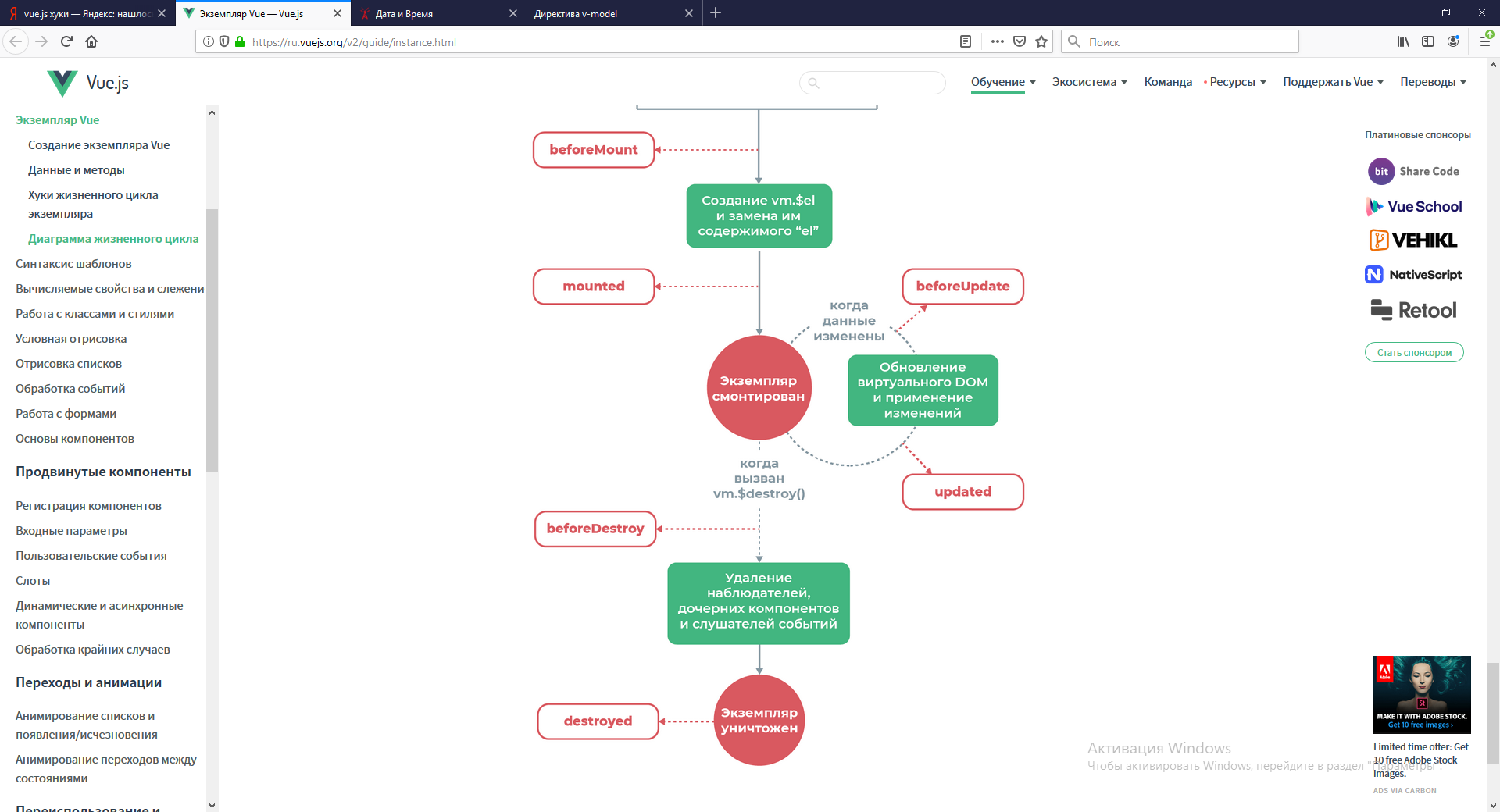
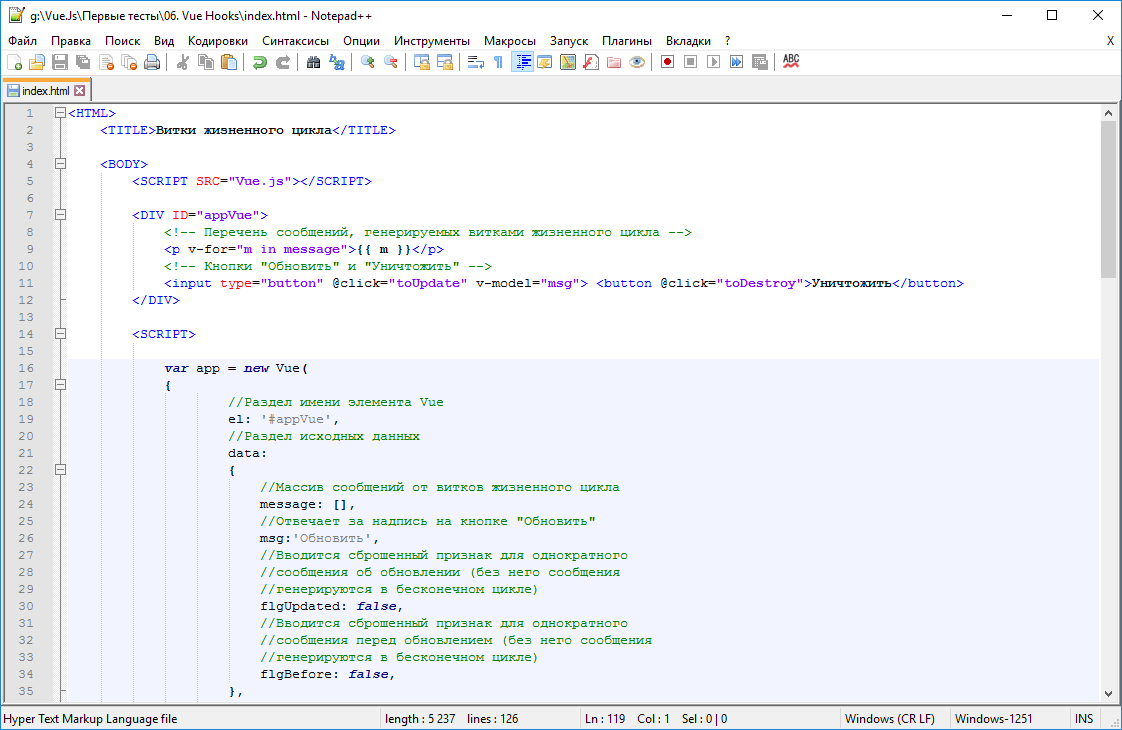


Рисунок 2 – Витки жизненного цикла экземпляра *Vue 2.x* (продолжение)



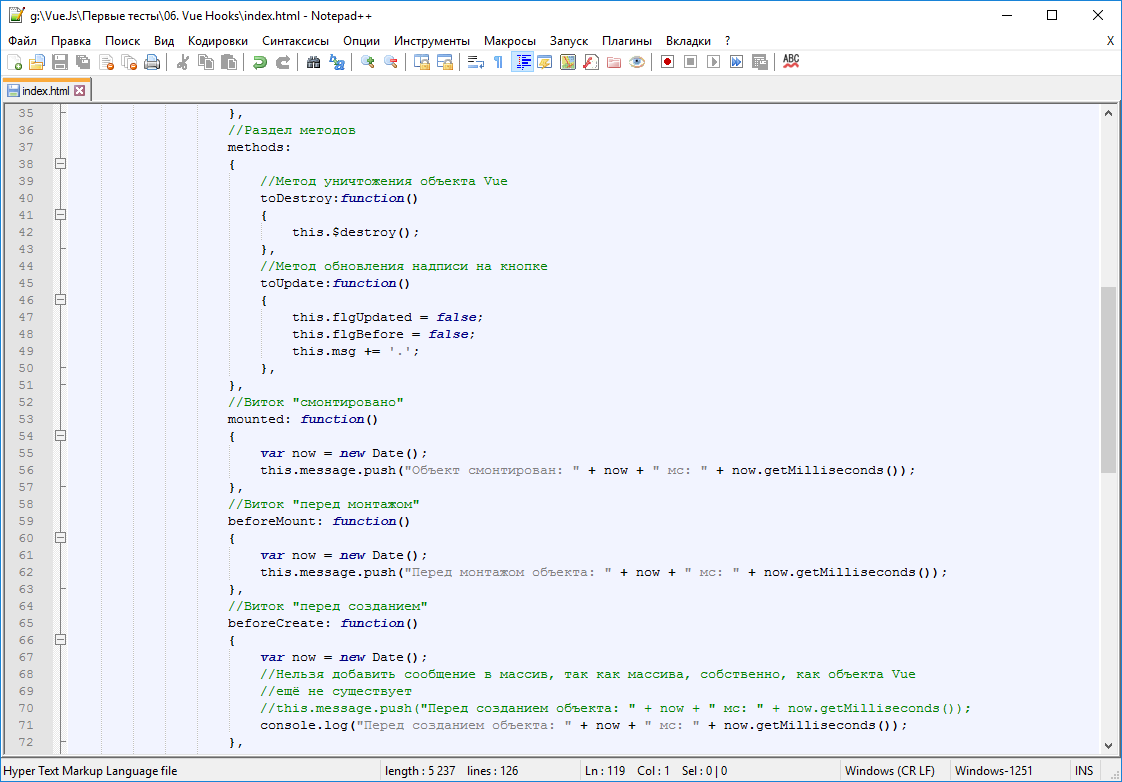
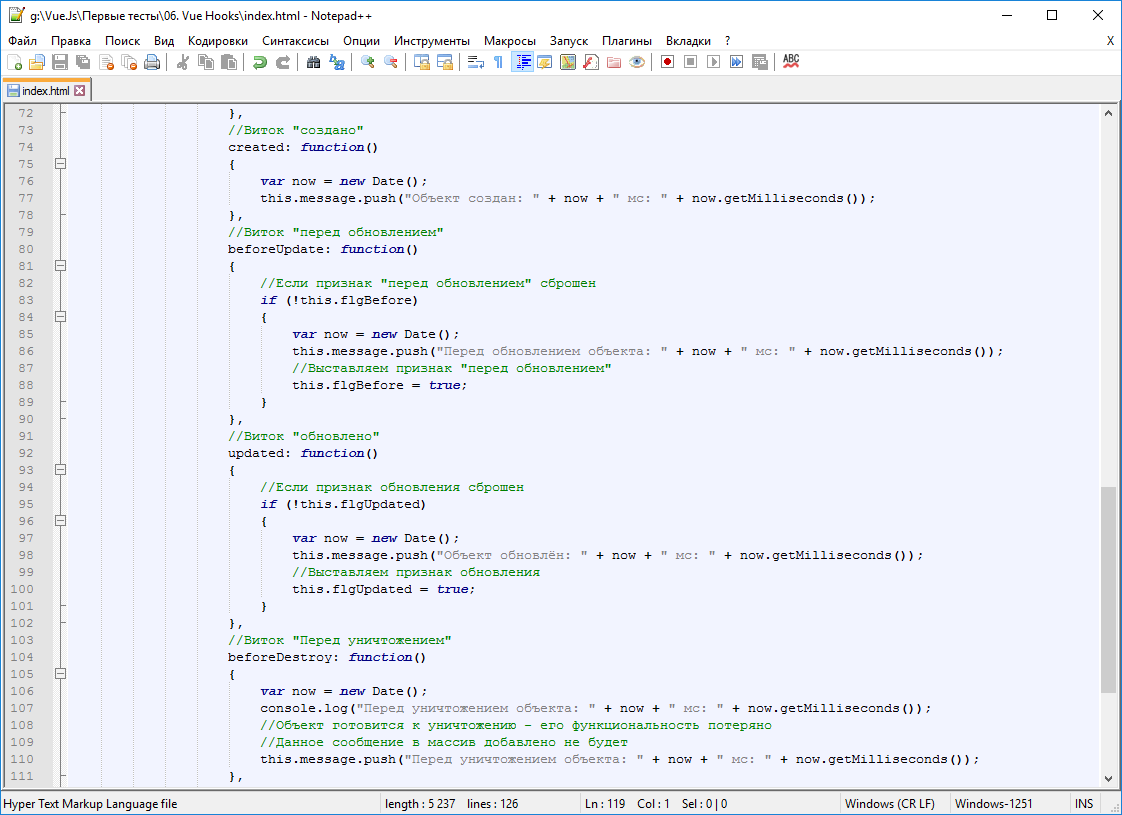


Рисунок 3 – Код примера тестирования работы витков ЖЦ *Vue 2.x* (начало)



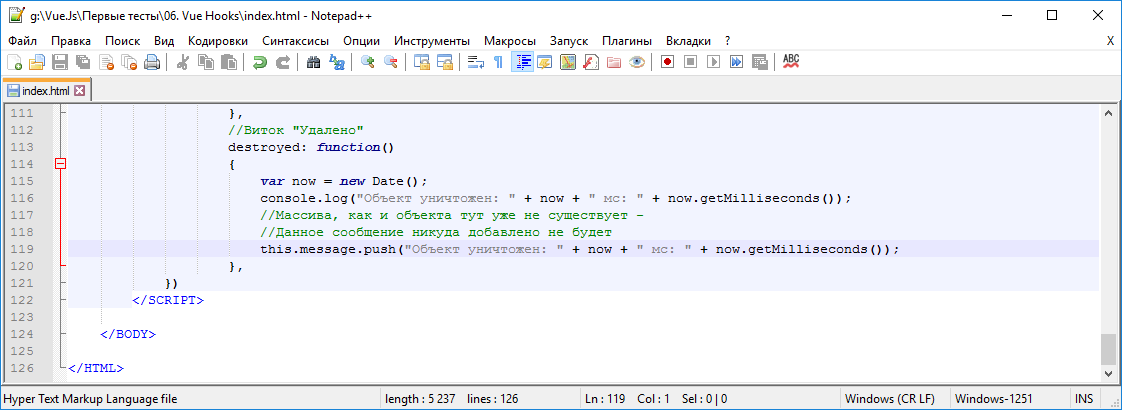


Рисунок 3 – Код примера тестирования работы витков ЖЦ *Vue 2.x* (продолжение)

**Приложение 3. Пример демонстрации версии браузера, используемого для выполнения работы*.***

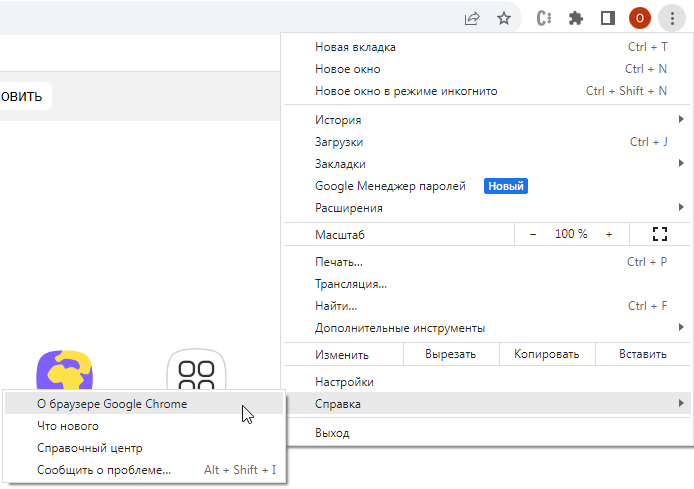


Рисунок 4 – Переход к получению сведений о браузере на примере *Google Chrome*

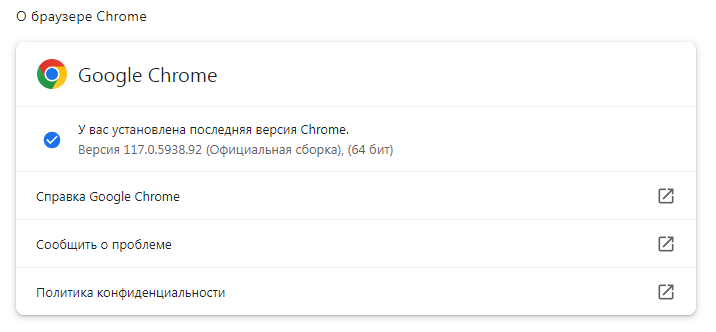


Рисунок 5 – Cведения о браузере *Google Chrome*

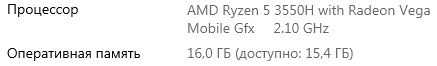
**Приложение 4. Структура таблицы соответствия переменных и методов, входящих в состав одностраничного *web*-приложения*.***

Таблица 1 – Пример таблицы соответствия переменных

|  |  |  |  |
| --- | --- | --- | --- |
| **№ п/п** | **Физический смысл конструкции** | **Наименование конструкции** | **Тип данных** |
| **1** | Метод инициации уничтожения экземпляра прототипа *Vue.js* | *toDestroy* | *function()* |
| **2** | Текущее состояние даты и времени | *now* | *Date()* |
| **3** | Динамическая надпись на экранной кнопке | *msg* | *string* |
| **4** | Список сообщений о событиях (журнал событий) | *message* | *string[]* |

**Приложение 5. Структура таблицы спецификации оборудования*.***

Например, в ОС семейства *Windows* эти сведения можно списать из раздела «Система».



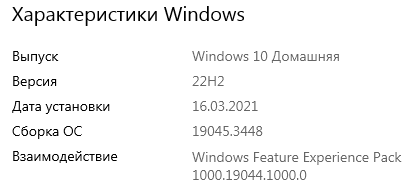


Рисунок 6 – Пример сведений об операционной системе

Таблица 2 – Пример таблицы спецификации параметров операционной системы

|  |  |  |
| --- | --- | --- |
| **№ п/п** | **Характеристика** | **Сведения** |
| **1** | Тип устройства | Стационарный ПК |
| **2** | Модель | Пользовательская сборка |
| **3** | Операционная система | *Microsoft Windows 10* |
| **4** | Процессор | *Intel Core i7* |
| **5** | Оперативная память | 8 Гб |
| **6** | Объём жёсткого диска | 1 Тб |