СОДЕРЖАНИЕ

[ВВЕДЕНИЕ 2](#__RefHeading___Toc13483931)

[1 Аналитический обзор поставленной задачи 3](#__RefHeading___Toc13483932)

[1.1 Обзор предметной области 3](#__RefHeading___Toc13483933)

[1.2 Постановка задачи 3](#__RefHeading___Toc13483934)

[2 Алгоритмическое конструирование 5](#__RefHeading___Toc13483935)

[2.1 Алгоритм работы программы 5](#__RefHeading___Toc13483936)

[2.2 Выводы 6](#__RefHeading___Toc13483937)

[3 Программное конструирование 7](#__RefHeading___Toc13483938)

[3.1 Выбор среды программирования 7](#__RefHeading___Toc13483939)

[3.2 Выбор среды разработки 8](#__RefHeading___Toc13483940)

[3.3 Основные модули программного средства 8](#__RefHeading___Toc13483941)

[3.4 Выводы 10](#__RefHeading___Toc13483942)

[4 Тестирование 11](#__RefHeading___Toc13483943)

[ЗАКЛЮЧЕНИЕ 13](#__RefHeading___Toc13483944)

[СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ 14](#__RefHeading___Toc13483945)

[ПРИЛОЖЕНИЕ А Исходный код программного средства 15](#__RefHeading___Toc13483946)

ВВЕДЕНИЕ

Алгоритмические задачи – задачи, которые решаются с помощью непосредственного применения определения, теоремы, то есть для решения которых имеется алгоритм.

В данном отчёт будет рассмотрен алгоритм решения задачи «Разработка программного модуля «датапикер» и модуля текстового представления отсчета времени», сформулированной в постановке задачи.

Целью программы является разработка алгоритма анализа действий пользователя и базового программного обеспечения.

Задачами модулей является улучшение пользовательского опыта.

В первой главе приведен обзор предметной области поставленной задачи, а также сформулировано ее условие.

Во второй главе представлен алгоритмы, используемые для решения поставленной задачи.

В третьей главе дано обоснование выбора языка программирования, представлены основные модули и функции программы.

В четвертой главе представлена работа созданного программного средства.

1 Аналитический обзор поставленной задачи

В данной главе будет рассмотрена актуальность темы работы и основные задачи программного средства. Также будет сформулирована постановка задачи.

1.1 Обзор предметной области

Модуль «Датапикер» — виджет графического интерфейса пользователя, который позволяет пользователю выбирать дату из календаря и / или время из временного диапазона. Обычной практикой является предоставление поля текстового поля, при нажатии которого для ввода даты открывается календарь рядом с полем или под ним, что позволяет пользователю заполнить поле соответствующей датой.

Модуль формирования текстового представления отсчета времени, которое можно использовать для удобного времени до какого-либо события (прим. «5 дней, 2 часа, 1 минута»).

**1.3 Постановка задачи**

Создать модули , соответствующее представленным выше требованиям. Модуль «Датапикер» должен предоставлять разработчику возможность встраивания компонента 2 видов: выбора одиночной даты, выбора диапазона дат. Также необходима возможность задавать компоненту начальное состояние.

Модуль формирования окончаний должен получать на вход числовую переменную — время отсчета до некоторого события в миллисекундах, на выходе необходимо получать это время в более удобном формате — информацию о том, сколько это в днях, часах и минутах.

2 Алгоритмическое конструирование

Для того чтобы алгоритм датапикера пришёл в действие, необходимо

вызвать событие «click» на соответствующем input-элементе.

Для использования модуля формирования окончаний необходимо передать в функцию 1 аргумента - число миллисекунд. Функция должна вернуть массив вида (n , m , v), где n — дни, m — часы, v — минуты.

2.1 Алгоритм работы программных модулей

Описание действия алгоритма «датапикера»:

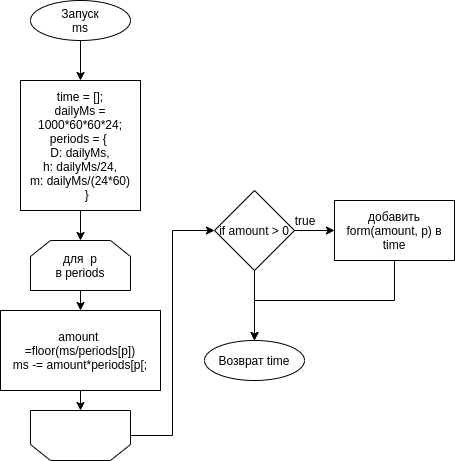
Отрисовывается стандартный input-элемент, который в будущем будет содержать выбранную дату или диапазон дат. При генерации события „click“ необходимо отрисовать 1 или 2 «календаря», содержащих: текстовое представление «YY:MM» на нем, 42 даты включая прошлый и следующий месяц в зависимости от дня недели последней даты текущего месяца, кнопки увеличения и уменьшения месяца.

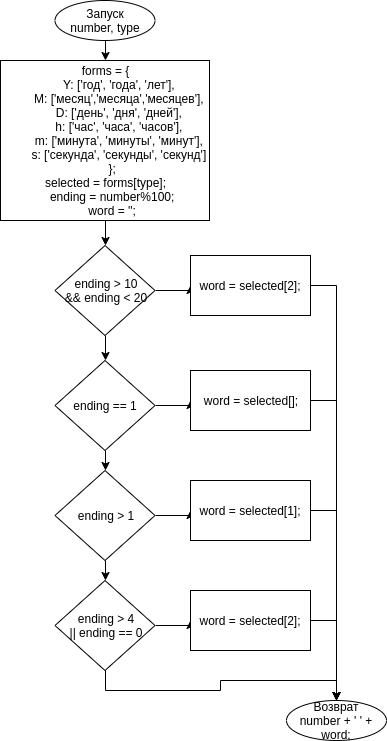
Пользователь может выбрать 1 дату или диапазон. При успешном выборе должен вызваться коллбэк с результатом выбора.

Действие алгоритма модуля формирования окончаний (см. Рисунок 1):

Шаг 1: Вызывается функция с 1 аргументом — число миллисекунд.

Шаг 2: Возвращается результат, полученные из модуля формирования окончаний(см. Рисунок 2) — массив вида (n, m , v), где n — дни, m — часы, v — минуты.

Рисунок 1. Схема формирования представления.

 Рисунок 2. Схема формирования окончаний.

2.2 Выводы

Разработаны два модуля для веб-сайта: модуль «датапикер», модуль формирования окончаний некоторых единиц времени.

Основной задачей разработкии является упрощение взаимодействия пользователя с веб-сайтом.

3. Программное конструирование

На основе выбранных алгоритмов решения поставленной задачи, выполнено программное конструирование, предполагающее выбор соответствующих требованиям сред разработки и программирования.

3.1 Выбор языка программирования

Для реализации разработанного алгоритма был выбран язык программирования JavaScript , фреймворк Vue.js.

Язык программирования JavaScript — [мультипарадигменный](https://ru.wikipedia.org/wiki/Мультипарадигменное_программирование) [язык программирования](https://ru.wikipedia.org/wiki/Язык_программирования). Поддерживает [объектно ориентированный](https://ru.wikipedia.org/wiki/Объектно-ориентированное_программирование), [императивный](https://ru.wikipedia.org/wiki/Императивное_программирование) и [функциональный](https://ru.wikipedia.org/wiki/Функциональное_программирование) стили. Является реализацией языка.

Язык программирования JavaScript обычно используется как встраиваемый язык для программного доступа к объектам [приложений](https://ru.wikipedia.org/wiki/Компьютерная_программа). Наиболее широкое применение находит в [браузерах](https://ru.wikipedia.org/wiki/Браузер) как язык сценариев для придания [интерактивности](https://ru.wikipedia.org/wiki/Интерактивность) [веб-страницам](https://ru.wikipedia.org/wiki/Веб-страница).

Основные архитектурные черты: [динамическая типизация](https://ru.wikipedia.org/wiki/Динамическая_типизация), [слабая типизация](https://ru.wikipedia.org/wiki/Слабая_типизация), [автоматическое управление памятью](https://ru.wikipedia.org/wiki/Сборка_мусора_(программирование)), [прототипное программирование](https://ru.wikipedia.org/wiki/Прототипное_программирование), функции как [объекты первого класса](https://ru.wikipedia.org/wiki/Объект_первого_класса).

Vue.js — JavaScript-фреймворк с открытым исходным кодом для создания пользовательских интерфейсов. Легко интегрируется в проекты с использованием других JavaScript-библиотек. Может функционировать как веб-фреймворк для разработки одностраничных приложений в реактивном стиле.

3.2 Выбор среды разработки

Среда Visual Studio Code — [редактор исходного кода](https://ru.wikipedia.org/wiki/Редактор_исходного_кода), разработанный [Microsoft](https://ru.wikipedia.org/wiki/Microsoft) для [Windows](https://ru.wikipedia.org/wiki/Windows), [Linux](https://ru.wikipedia.org/wiki/Linux) и [macOS](https://ru.wikipedia.org/wiki/MacOS). Позиционируется как «лёгкий» редактор кода для [кроссплатформенной](https://ru.wikipedia.org/wiki/Кроссплатформенность) разработки [веб](https://ru.wikipedia.org/wiki/Веб-приложение)- и [облачных](https://ru.wikipedia.org/wiki/Облачные_вычисления) приложений. Включает в себя [отладчик](https://ru.wikipedia.org/wiki/Отладчик), инструменты для работы с [Git](https://ru.wikipedia.org/wiki/Git), [подсветку синтаксиса](https://ru.wikipedia.org/wiki/Подсветка_синтаксиса), [IntelliSense](https://ru.wikipedia.org/wiki/IntelliSense) и средства для [рефакторинга](https://ru.wikipedia.org/wiki/Рефакторинг). Имеет широкие возможности для [кастомизации](https://ru.wikipedia.org/wiki/Кастомизация): пользовательские темы, [сочетания клавиш](https://ru.wikipedia.org/wiki/Сочетание_клавиш) и [файлы конфигурации](https://ru.wikipedia.org/wiki/Файл_конфигурации). Распространяется бесплатно, разрабатывается как программное обеспечение с [открытым исходным кодом](https://ru.wikipedia.org/wiki/Открытое_программное_обеспечение), но готовые сборки распространяются под [проприетарной лицензией](https://ru.wikipedia.org/wiki/Проприетарное_программное_обеспечение).

Visual Studio Code основан на [Electron](https://ru.wikipedia.org/wiki/Electron) — [фреймворк](https://ru.wikipedia.org/wiki/Фреймворк), позволяющий с использованием [Node.js](https://ru.wikipedia.org/wiki/Node.js) разрабатывать настольные приложения, которые работают на [движке Blink](https://ru.wikipedia.org/wiki/Blink_(движок)). Несмотря на то, что редактор основан на [Electron](https://ru.wikipedia.org/wiki/Electron), он не использует редактор [Atom](https://ru.wikipedia.org/wiki/Atom_(текстовый_редактор)). Вместо него реализуется веб-редактор Monaco, разработанный для Visual Studio Online.

3.3 Основные модули программного средства

Файловая структура модуля «датапикер» состоит из файлов calendar.vue с логикой работы и вёрсткой календаря, DatePicker.vue с кодом самого вёрсткой и логикой работы самого модуля, вызове события «click» вне на модуля.

Файловая структура модуля формирования окончаний состоит из исполняемого asd.js файла с 2 функциями (см. таблицу 1).

Реализованный компонент calendar принимает входные параметры:

* month – число, year – число, определяющие год и месяц на календаре.
* selected – массив объектов Date или объект Date (выбранные даты).
* onChange – функция, вызывающаяся при событии “click” на дате, onMonthChange – функция, вызывающаяся при смене месяца.
* disableNext и disablePrev – bool параметры отвечающией за отображение кнопок увеличения и уменьшения месяца на календаре.

Файл также содержит в себе верстку(<template/>) и стили(<style/>).

Реализованный компонент DatePicker принимает входные параметры:

* range – bool , определяет режим работы (подбор 1 даты или диапазона).
* selected – массив объектов Date или объект Date (выбранные даты).
* onSubmit – функция, вызывающаяся после выбора даты или диапазона.

Файл также содержит в себе верстку(<template/>) и стили(<style/>).

Реализованная директива outside принимает всего один параметр – функцию, которая будет вызвана при событии “click” вне компонента, который использует outside.

Таблица 1. Файл asd.js

|  |  |  |
| --- | --- | --- |
| Название Функции | Входные параметры | Результат |
| form(number, type) | number,type | Данная функция формирует number — число мсек. В единицу времени type. |
| ff(number) | number | Данная функция функция для number — числа миллисекунд вернет информацию о том, сколько это в днях, часах, минутах. |

3.4 Выводы

Реализована программа, позволяющая решить поставленную задачу на основании выбора пользователя. Основные требования реализованы, модули протестироаны и к использованию в настоящем проекте.

4 Тестирование

До начала взаимодействия «Датапикер» выглядит как обычный input (рис. 1).



Рисунок 1-Старт

На рисунке 2 представлен компонент до выбора дат.

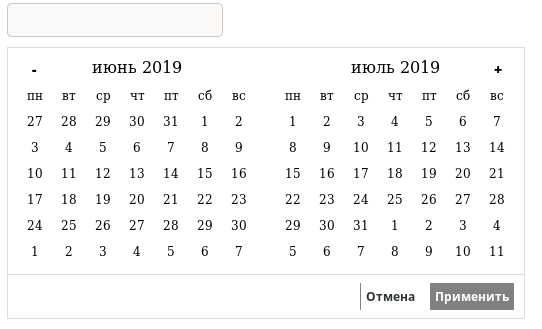


Рисунок 2-Компонент до выбора

На рисунке 3 представлен вид после выбора диапазона.

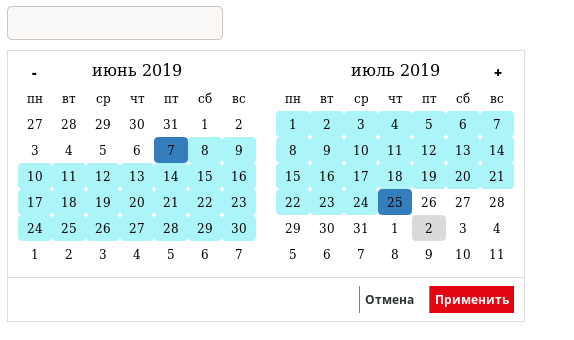
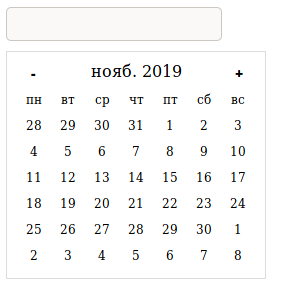


Рисунок 3 - Выбран диапазон

На рисунке 4 представлен компонент с возможностью выбора только одной даты.

Рисунок 4 - Выбор одной даты

Результаты проведенного тестирования показали, что разработанное программное средство в полной мере выполняет поставленную задачу, а также корректно реагирует на любые действия пользователя, обрабатывая все возможные исключения.

**ЗАКЛЮЧЕНИЕ**

В рамках данной работы было создано 2 модуля для веб-сайта.

В ходе выполнения работы был построен аналитически обоснованный алгоритм решения поставленной задачи и разработано соответствующее программное средство.

# СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Дэвид Макфарланд JavaScript и jQuery. Исчерпывающее руководство : Эксмо, 2017. – 880 с.
2. Орлов, С. Теория и практика языков программирования. Учебник для вузов Стандарт третьего поколения / С. Орлов. - СПб.: Питер, 2017. - 688 c
3. Давыдова, Н. А. Программирование [Электронный ресурс] : учебное пособие / Н. А. Давыдова, Е. В. Боровская. - 2 изд. (эл.). - М. : БИНОМ. Лаборатория знаний, 2012. - 238 с. : ил. - (Педагогическое образовании.

ПРИЛОЖЕНИЕ А

**Исходный код программного средства**

Листинг А.1 - Исходный код файла calendar.vue

<template>

<div class="calendar">

<div class="header">

<button v-if="!disablePrev" @click="decMonth" class="day btn prev hover">-</button>

<span class="title">{{ [lang.shortMonths[state.getMonth()], state.getFullYear()].join(' ') }}</span>

<button v-if="!disableNext" @click="incMonth" class="day btn next hover">+</button>

</div>

<div class="days-header">

<div class="day" v-for="n in 7" :key="n">{{lang.shortWeekDays[n-1]}}</div>

</div>

<div class="days-container">

<div v-for="n in 42"

@click="handleDateClick(offset(n))"

:key="offset(n)"

class="day hover"

:class="{selected: isSelected(offset(n)), inrange: isInRange(offset(n))}">

{{ dateNumber(offset(n)) }}

</div>

</div>

</div>

</template>

<script>

export default {

name: 'Calendar',

data: function() {

return {

state: this.year && this.month ? new Date(this.year, this.month) : new Date(),

}

},

watch: {//Нужно для синхронизации двух календарей, без этого не обновляется.

month: function(newMonth) {

this.state = new Date(this.year, newMonth);

},

year: function(newYear) {

this.state = new Date(newYear, this.month);

},

},

computed: {

\_year: function() {

return this.state.getFullYear();

},

\_month: function() {

return this.state.getMonth();

},

\_selected: function() {

return [].concat(this.selected).filter(x => x instanceof Date);

},

},

methods: {

//Функции для календаря

incMonth: function() {

this.state = new Date(this.state.getFullYear(), this.state.getMonth() + 1);

this.onMonthChange(1);//test

},

decMonth: function() {

this.state = new Date(this.state.getFullYear(), this.state.getMonth() - 1);

this.onMonthChange(-1);//test

},

//Функции для дат на календаре;

isSelected: function(offset) {

if (this.\_selected.length == 0) return;

let date = this.offsetToDate(offset);

let isSelected = this.\_selected.some(x => {

return x.getFullYear() == date.getFullYear() && x.getMonth() == date.getMonth() && date.getMonth() == this.state.getMonth() && x.getDate() == date.getDate();

});

if (isSelected) return true;

return false;

},

isInRange: function(offset) {

if (this.\_selected.length != 2) return;

let date = this.offsetToDate(offset);

let sortedSelects = [].concat(this.\_selected).sort((x,y) => x-y);

let isInRange = date < sortedSelects[1] && date > sortedSelects[0] && date.getMonth() == this.state.getMonth();

if (isInRange) return true;

return false;

},

handleDateClick: function(offset) {

let date = this.offsetToDate(offset);

if (date.getMonth() > this.state.getMonth()) this.incMonth();

else if (date.getMonth() < this.state.getMonth()) this.decMonth();

//

this.onChange(date);

},

dateNumber: function(offset) {

let date = this.offsetToDate(offset);

return date.getDate();

},

offset: function(n) {

let date = new Date(this.state.getFullYear(), this.state.getMonth());

date.setDate(1);

let day = date.getDay();

let daysPrevMonth = 0;

if (day == 0) daysPrevMonth = 6;

else daysPrevMonth = day - 1;

return n - daysPrevMonth;

},

offsetToDate: function(offset) {

let date = new Date(this.state.getFullYear(), this.state.getMonth());

date.setDate(offset);

return date;

},

},

props: {

year: Number,

month: Number,

selected: [Array, Date],

onChange: {

type: Function,

required: true,

},

onMonthChange: {

type: Function,

default: () => {}

},

disablePrev: Boolean,

disableNext: Boolean,

lang: {

type: Object,

default: () => ru,

},

},

}

</script>

<style scoped>

.days-container {

display: flex;

padding: 0;

margin: 0;

list-style-type: none;

flex-direction: row;

flex-wrap: wrap;

}

.day {

display: inline-block;

white-space: nowrap;

text-align: center;

vertical-align: middle;

min-width: 32px;

width: 32px;

height: 24px;

line-height: 24px;

font-size: 12px;

border-radius: 4px;

border: 1px solid transparent;

white-space: nowrap;

cursor: pointer;

color: black;

}

.calendar {

display: inline-block;

width: 238px;

max-width: 238px;

padding: 10px;

}

.header {

display: flex;

justify-content: center;

}

.title {

position: absolute;

margin: auto;

}

.btn {

font-weight: 1000;

font-size: 14px;

border: 0;

background: 0;

color: 0;

outline: 0;

}

.prev {

margin-right: auto;

}

.next {

margin-left: auto;

}

.hover:hover {

background-color: #DADADA;

}

.other {

color: grey;

}

.selected {

background: #357ebd !important;

}

.inrange {

background: #abf4f8;

}

</style>

Исходный код файла DatePicker.vue

<template>

<div ref="dpRef" v-outside:[setDisplayFalse] class="datepicker">

<input ref="inputRef" @click="ff" @click.prevent="displayModal=!displayModal" :value="submited.map(x=>x.toLocaleDateString()).join(' - ')" type="text" name="datepicker" readonly/>

<div v-if="displayModal" class="modal" :style="{top: (offsetTop + inputHeight + 10) + 'px', left: (offsetLeft) + 'px' }" >

<template v-if="range">

<Calendar

:onMonthChange="handleMonthChange"

:onChange="handleRangeCalendarDateSelect"

:selected="m\_selected"

:year="\_year"

:month="\_month"

disableNext/>

<Calendar

:onMonthChange="handleMonthChange"

:onChange="handleRangeCalendarDateSelect"

:selected="m\_selected"

:year="\_year"

:month="\_month + 1"

disablePrev/>

</template>

<template v-else>

<Calendar :year="\_year"

:month="\_month"

:selected="m\_selected"

:onChange="handleSingleCalendarDateSelect"

:onMonthChange="handleMonthChange"/>

</template>

<div v-if="range" class="footer">

<span class="result">{{submited.map(x=>x.toLocaleDateString()).join(' - ')}}</span>

<button @click="handleReturn" class="close">Отмена</button>

<button :disabled="range && m\_selected.length != 2" @click="handleSubmit" class="submit">Применить</button>

</div>

</div>

</div>

</template>

<script>

import Calendar from './Calendar.vue'

import outside from '../directives/clickoutside.vue'

export default {

name: 'DatePicker',

components: {

Calendar

},

directives: {

outside

},

data: function() {

return {

m\_selected: [].concat(this.selected).filter(x => x instanceof Date),

state: this.initState ? this.selected[0] : new Date(),

submited: [],

displayModal: false,

//hovered: null,

}

},

watch: {

},

methods: {

ff: function() {

console.log(this.offsetLeft, this.offsetTop);

},

incMonth: function() {

this.state = new Date(this.\_year, this.\_month + 1);

},

decMonth: function() {

this.state = new Date(this.\_year, this.\_month - 1);

},

handleMonthChange: function(diff) {

if (diff > 0) this.incMonth();

else this.decMonth();

},

handleSubmit: function() {

this.submited = this.m\_selected;

this.displayModal = false;

this.onSubmit(this.submited);

},

handleReturn: function() {

this.m\_selected = [];

this.displayModal = false;

this.submited = [];

},

handleRangeCalendarDateSelect: function(date) {

if (this.m\_selected.length == 2) this.m\_selected = [date];

else if (this.m\_selected.length == 1) {

if (this.m\_selected[0].getTime() > date.getTime()) this.m\_selected = [date];

else this.m\_selected = this.m\_selected.concat(date);

}

else this.m\_selected = this.m\_selected.concat(date);

},

handleSingleCalendarDateSelect: function(date) {

this.m\_selected = [date];

this.handleSubmit();

},

setDisplayFalse: function() {

this.displayModal = false;

}

},

computed: {

offsetLeft: function() {

return this.$refs.dpRef.offsetLeft;

},

offsetTop: function() {

return this.$refs.dpRef.offsetTop;

},

inputHeight: function() {

return this.$refs.inputRef.offsetHeight;

},

\_year: function() {

return this.state.getFullYear();

},

\_month: function() {

return this.state.getMonth();

},

},

props: {

range: Boolean,

selected: [Array, Date],

onSubmit: {

type: Function,

default: () => {},

}

}

}

</script>

<style scoped>

.datepicker {

display: inline-block;

}

.result {

padding-right: 10px;

}

.modal {

border: 1px solid #dddddd;

position: fixed;

}

.footer {

text-align: right;

border-top: 1px solid #dddddd;

padding: 8px 0;

}

button {

border: 0;

background: 0;

outline: 0;

padding: 0;

font-size: 12px;

font-weight: 550;

display: inline-block;

border-left: 1px solid grey;

padding: 5px;

}

button:hover {

outline: 1px solid green;

}

button:active {

outline: 1px solid white;

}

button:disabled {

outline: none;

border: none;

background: gray;

}

.submit {

color: white;

font-weight: 550;

background: #e30611;

margin: 0 10px;

}

</style>

Исходный код файла outside.vue

<script>

let handleClick;

export default {

name: 'outside',

bind: function(el, binding) { // function(el, binding, vnode)

if (typeof binding.arg == 'function') {

handleClick = (event) => {

if (!el.contains(event.target)) binding.arg();

}

}

else handleClick = () => {console.log("argument isn't callback-function")}

document.addEventListener('click', handleClick);

},

unbind: function() { //function(el, binding, vnode)

document.removeEventListener('click', handleClick);

}

}

</script>

Исходный код файла asd.js

var form = function(number, type) {

var forms = {

Y: ['год', 'года', 'лет'],

M: ['месяц','месяца','месяцев'],

D: ['день', 'дня', 'дней'],

h: ['час', 'часа', 'часов'],

m: ['минута', 'минуты', 'минут'],

s: ['секунда', 'секунды', 'секунд']

};

var selected = forms[type];

var ending = number%100;

var word = '';

if (ending > 10 && ending < 20) word = selected[2];

else {

ending %= 10;

if (ending == 1) word = selected[0];

if (ending > 1) word = selected[1];

if (ending > 4 || ending == 0) word = selected[2];

}

return number + ' ' + word;

}

var ff = function(ms) {

var time = [];

var dailyMs = 1000\*60\*60\*24;

var periods = { //Имена полей строго как в form, в порядке возрастания;

D: dailyMs,

h: dailyMs/24,

m: dailyMs/(24\*60)

}

for (var p in periods) {

var amount = Math.floor(ms/periods[p]);

if (amount > 0) time.push(form(amount, p));

ms -= amount\*periods[p];

}

return time

}