**Министерство образования Республики Беларусь**

**Учреждение образования**

**Международный государственный экологический университет**

**имени А. Д. Сахарова БГУ**

Факультет мониторинга окружающей среды

Кафедра экологических информационных систем

**ИССЛЕДОВАНИЕ ТЕХНОЛОГИИ D3.JS НА ПРИМЕРЕ СОЗДАНИЯ WEB-ПРИЛОЖЕНИЯ, ВИЗУАЛИЗИРУЮЩЕГО СТАТИСТИКУ ЗДРАВООХРАНЕНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ.**

Курсовой проект студента 5-го курса

ЖУРАВКИНА Дмитрия Владимировича

\_\_\_\_\_\_\_\_\_\_\_\_Д. В. Журавкин

|  |  |
| --- | --- |
| «Допустить к защите»  Зав. кафедрой экологических информационных систем,  к.ф.-м.н., доцент  \_\_\_\_\_\_\_\_\_\_\_\_\_\_ В. А. Иванюкович  «\_\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_2017 г. | Научный руководитель  Преподаватель кафедры ЭИС  \_\_\_\_\_\_\_\_\_\_\_И. В. Лефанова |

Минск 2017

НАЗВАНИЕ РАБОТЫ

*Рефе**рат*

Курсовая работа 21 с.: 4 рис.,41 источник.

ИССЛЕДОВАНИЕ ТЕХНОЛОГИИ D3.JS НА ПРИМЕРЕ СОЗДАНИЯ WEB-ПРИЛОЖЕНИЯ, ОТРАЖАЮЩЕГО СТАТИСТИКУ ЗДРАВООХРАНЕНИИЯ В РЕСПУБЛИКЕ БЕЛАРУСЬ.

Главным объектом исследования является библиотека языка программирования JavaScript D3.js.

Второстепенным объектом исследования выступает статистика здравоохранения Республики Беларусь.

Цель работы­освоить инструмент визуализации данных D3.js, исследовать тенденции развития современных web-приложений.

Для достижения цели необходимо решить следующие задачи: отобрать и отформатировать данные здравоохранения, создать архитектуру приложения и выбрать инструменты для её реализации.

*РЕФЕРАТ на белорусском языке*

*Рэферат*

Курсавая работа 21 старонка, 5 малюнакау, 41 крынiца**.**

ДАСЛЕДАВАННЕ ТЭХНАЛОГІІ D3.JS НА ПРЫКЛАДЗЕ СТВАРЭННЯ WEB-ПРЫЛАЖЭНIЯ, ЯКОЕ АДЛЮСТРОВЫВАЕ СТАТЫСТЫКУ АХОВЫ ЗДАРОУЯ У РЭСПУБЛИЦЫ БЕЛАРУСЬ.

Галоўным аб'ектам даследавання з'яўляецца бібліятэка мовы праграмавання JavaScript D3.js.

Дадатковым аб'ектам даследавання выступае статыстыка аховы здароўя Рэспублікі Беларусь.

Мэта работы¬вывучыць інструмент візуалізацыі дадзеных D3.js, даследаваць тэндэнцыі развіцця сучасных web-прыкладанняў.

Для дасягнення мэты неабходна вырашыць наступныя задачы: адабраць і адфарматаваць дадзеныя аховы здароўя, стварыць архітэктуру прыкладання і выбраць інструменты для яе рэалізацыі.

*РЕФЕРАТ на английском языке*

*Abstract*

Graduate work 21 pages, 5 figure, 41 references**.**

STUDY OF TECHNOLOGY D3.JS ON THE EXAMPLE OF CREATING WEB-APPLICATION, REFLECTING STATISTICS OF PUBLIC HEALTH IN THE REPUBLIC OF BELARUS.

The main object of research is the JavaScript programming language library D3.js.

The secondary object of research is health statistics of the Republic of Belarus.

The purpose of the work is to master the data visualization tool D3.js, to investigate the development trends of modern web-applications.

To achieve the goal it is necessary to solve the following tasks: select and format health data, create an application architecture and choose tools for its implementation.

**Содержание**

[Введение 4](#_Toc453345628)

[1 Разработка эскизного и технического проектов программы 5](#_Toc453345629)

[1.1 Назначение и область применения 5](#_Toc453345630)

[1.2 Технические характеристики 5](#_Toc453345631)

[2 Разработка рабочего проекта 7](#_Toc453345632)

[2.1 Разработка проекта 7](#_Toc453345633)

[2.2 Спецификация программы 8](#_Toc453345634)

[2.3 Текст программы(основной алгоритм) 8](#_Toc453345635)

[2.4 Описание программы 12](#_Toc453345636)

[2.5 Тестирование 13](#_Toc453345637)

[3 Руководство оператора 14](#_Toc453345638)

[Заключение 17](#_Toc453345639)

[Список использованных источников 18](#_Toc453345640)

# Введение

Сегодня WEB – самая быстрорастущая платформа программного обеспечения в мире. Объёмы данных стремительно увеличиваюся в месте с необходимостью их качественно обрабатывать и позиционировать.

Графическая информация - наиболее удобная форма для восприятия человеком. Графика делиться на растровую (jpg, png) и векторную (svg). Преимущество последней заключается в её масштабируемости. Векторная графика принимает любые объёмы без потери качества потому что вычисляется математическими формулами, а не храниться в виде координат пикселей. Это особенность векторной графики делает её очень привлекательной для использования.

Одним из лучших инструментом программирования векторной графики в вебе, является JavaScript библиотека D3.js. Она имеет широкое распространение и большой набор возможностей. По этому я решил остановить свой выбор на ней.

# 1 Разработка эскизного и технического проектов программы

## **1.1 Назначение и область применения**

Назначение – справочная информация. Данное веб­приложение предназаначено для удобного, доступного, кросплатформенного и кросбраузерного информирования пользователей сети интернет по теме здравоохранения в Республике Беларусь. Изучение предоставленых материалов, способствует повышению медицинской грамотности населения.

## **1.2 Технические характеристики**

Программа состоит из 3 основных файлов и 15 вспомагательных. Общий объём не превышает 4 Mb.

### 1.2.1 Постановка задачи

Изучить современные стандарты построения веб-приложений. Использовать их преимущества. Найти нужные данные по здравоохранению Республики Беларусь. Отобрать и сформировать их. Создать бизнес логику приложения. Реализовать удобный и интуитивно понятный пользовательский интерфейс.

### 1.2.2 Выбор состава технических и программных средств

Реализация проекта осуществляется в операционной системе Windows 7 и Windows 8 (на ноутбуке).

Основная программа для работы тектовый редактор ATOM. Он просто, надёжный, имеет оптимальный набор инструментов и глубоко продуманный контроль горячих клавиш. При необходимости существует доступ к многичесленным расширениям, которые помогают улучшить и оптимизировать рабочий процесс. Из использованных расширений используется emmet, color picker, git и т.д..

Логическое ядро написано на языке программирования JavaScript с использованием его последних стандартов ECMAScript2015 (ES6). Большинство кода использует фунции библиотеки визуализации данных D3.js..

Для компиляции CSS был выбран препроцессор SASS т.к. он имеет самое широкое сообщество. SASS позволяет использовать функции недоступные в самом CSS, например, переменные, вложенности, миксины, наследование и другие приятные вещи, расширяющие удобство написания CSS.

Для сборки проекта, контроля версий и линковки зависимостей, использует сборщик WebPack2.0.

# 2 Разработка рабочего проекта

### 2.1 Разработка проекта

Разработка проекта начинается с поиска актуальной и достоверной информации. Моё внимание останавливается на сайте национального статистического коммитета Республики Беларусь [www.belstat.gov.by](http://www.belstat.gov.by). Там я нашёл некоторые данные здравоохранения с 2000 по 2015 гг..

Проектирование архитектуры системы заняло много времени. Было решено визуализировать данные тремя способами представления информации: гистограмма, график и поток.

Данные были перенесены в код программы вручную, образуя массив объектов JavaScript.

Широко использовалась библиотека D3. Она помогла создать оси координат, а поставив на них обработчики событий, мне удалось обновлять и масштабировать оси, в зависимости от требуемых данных. Так же, широко использовались функции выделения элементов DOM, назначения им различных классов и атрибутов.

### 2.2 Спецификация программы

Начальная часть программы это список используемых переменных. Туда входят сами данные (массив объектов data), массив цветов. Далее идут переменные отвечающие за размеры графики, базирующиеся на константе ширины экрана пользователя. За ними следуют специфические константы библиотеки d3, такие как масштабы, координатные оси, зумы.

Главная часть программы это обрабочких событий кликов меню. Это совокупность следующих функций: change – которая определяет форму визуализации и вызывает функцию render. Функция toggleActive переключает классы css для пунктов меню, включая и выключая активные. Функция Render – высчитывает набор активных кнопок меню и вызывает одну из трёх функций рисования, в зависимости от выбора пользователя.

## 2.3 Текст программы(основной алгоритм)

Текст разметки html и стилей css решено не вносить, чтобы не перегружать данную пояснительную записку. Ниже представлен основной алгоритм работы программы, на языке JavaScript.

const CONTAINER\_W = parseFloat(d3.select('container').style('width'));

const margin = {top: 20, right: 30, bottom: 20, left: 70},

width = CONTAINER\_W - margin.left - margin.right,

height = CONTAINER\_W / 2 - margin.top - margin.bottom - 90;

const svg = d3.select("svg")

.attr("width", width + margin.left + margin.right)

.attr("height", height + margin.top + margin.bottom)

.append("g")

.attr("transform", "translate(" + margin.left + "," + margin.top + ")");

const layout = svg.append('g');

//--------------SCALES---------------------------------------

const xScale = d3.scaleTime()

.domain([new Date(2000,01,01), new Date(2016,01,01)])

.range([0, width]);

const yScale = d3.scaleLinear()

.range([height, 0])

.domain([0, 125000]);

const xBandWide = d3.scaleBand()

.rangeRound([0, width])

.paddingInner(0.1)

.domain(data.map(e=>e.year.slice(0,4)));

const xBandNarrow = d3.scaleBand()

.padding(0.05)

.domain(Object.keys(colors))

.rangeRound([0, xBandWide.bandwidth()]);

//--------------AXES----------------------------------------

const xx = d3.axisBottom(xScale)

.ticks(width / height \* 5);

const yy = d3.axisLeft(yScale)

.ticks(width / height \* 5)

.tickSizeInner(width);

// const xxBandW = d3.axisBottom(xBandWide)

// .ticks(width / height \* 5);

const xxBandN = d3.axisBottom(xBandNarrow)

.ticks(width / height \* 5);

var axisXnarrow;

const axisX = svg.append("g")

.attr("transform", "translate(0, " + height + ")")

.call(xx);

let axisY = svg.append("g")

.attr("transform", "translate(" + width + ", 0)")

.call(yy);

axisY.select(".domain").remove();

axisY.selectAll(".tick:not(:first-of-type) line")

.attr("stroke", "#777")

.attr("stroke-dasharray", "2,2");

//------------ZOOM-------------------------------------------

const zoom = d3.zoom()

.scaleExtent([-10, 10])

.translateExtent([[-5, -5], [width + 10, height + 10]])

.on('zoom', zoomed);

layout.call(zoom);

function zoomed() {

layout.attr('transform', d3.event.transform);

axisX.call(xx.scale(d3.event.transform.rescaleX(xScale)));

axisY.call(yy.scale(d3.event.transform.rescaleY(yScale)));

axisY.selectAll(".tick:not(:first-of-type) line")

.attr("stroke", "#777")

.attr("stroke-dasharray", "2,2");

};

//-------------RENDER----------------------------------------------

d3.selectAll('input.group1').on('change', change);

d3.selectAll('input.group2').on('click', toggleActive);

d3.selectAll('input.group2').on('change', render);

let view = 'radio-2';

function change() {

view = d3.select(this).attr('id');

render();

}

function toggleActive() {

d3.select(this).classed('active', !d3.select(this).classed('active'));

let label = d3.select(`label[for=${this.id}]`);

let value = this.value;

d3.select(this).classed('active')

? label.style('background', colors[value])

: label.style('background', null);

}

function render() {

layout.selectAll('path').remove();

layout.selectAll('.hist').remove();

let activeTabs = d3.selectAll('.active').\_groups[0];

activeTabs = Array.from(activeTabs).map(e => e.value);

let reqFun = (view == 'radio-1') ? drawHist

: (view == 'radio-2') ? drawPath

: drawStream;

if(reqFun == drawPath){

for(let i = 0; i < activeTabs.length; i++){

reqFun(activeTabs[i]);

}

}

else if (reqFun == drawStream) {

reqFun(activeTabs);

}

else if (reqFun == drawHist) {

reqFun(activeTabs);

}

}

//---------------------DRAW FUNCS----------------------------------------

function drawHist(activeTabs) {

let domain = activeTabs.map(e => d3.max(data, d => d[e]));

domain = d3.max(domain);

yScale.domain([0, domain]);

axisY.remove();

axisY = svg.append("g")

.attr("transform", "translate(" + width + ", 0)")

.call(yy);

axisY.select(".domain").remove();

axisY.selectAll(".tick:not(:first-of-type) line")

.attr("stroke", "#777")

.attr("stroke-dasharray", "2,2");

// axisXnarrow = svg.append("g")

// .attr("transform", `translate(0,${height-20})`)

// .call(xxBandN)

// .selectAll('text').remove();

layout.selectAll('g')

.data(data)

.enter().append('g')

.classed('hist', true)

.attr('transform', function(d){

return `translate(${xScale(new Date(d.year))}, 0)`})

.selectAll('rect')

.data(d => Object.keys(colors).map((key) => {

return {key: key, value: d[key]}

}))

.enter().append('rect')

.attr('x', d => xBandNarrow(d.key))

.attr('y', d => 50)

.attr('width', xBandNarrow.bandwidth())

.attr('heigth', d => height - yScale(d.value))

.attr('fill', d => colors[d.key]);}

function drawPath(value) {

let activeTabs = d3.selectAll('.active').\_groups[0];

activeTabs = Array.from(activeTabs).map(e => e.value);

let domain = activeTabs.map(e => d3.max(data, d => d[e]));

domain = d3.max(domain);

yScale.domain([0, domain]);

axisY.remove();

axisY = svg.append("g")

.attr("transform", "translate(" + width + ", 0)")

.call(yy);

axisY.select(".domain").remove();

axisY.selectAll(".tick:not(:first-of-type) line")

.attr("stroke", "#777")

.attr("stroke-dasharray", "2,2");

let line = d3.line()

.x(d => xScale(new Date(d.year)))

.y(d => yScale(d[value]))

.curve(d3.curveBasis);

layout.append('path')

.datum(data)

.attr('fill', 'none')

.attr('stroke', d => colors[value])

.attr('stroke-width', 3)

.attr('stroke-linejoin', 'round')

.attr('stroke-linecap', 'round')

.attr('id', value)

.attr('d', line);

}

function drawStream(activeTabs) {

let domain = activeTabs.map(e => d3.max(data, d => d[e]));

domain = d3.sum(domain);

yScale.domain([0, domain]);

axisY.remove();

axisY = svg.append("g")

.attr("transform", "translate(" + width + ", 0)")

.call(yy);

axisY.select(".domain").remove();

axisY.selectAll(".tick:not(:first-of-type) line")

.attr("stroke", "#777")

.attr("stroke-dasharray", "2,2");

let area = d3.area()

.x(function(d) {return xScale(new Date(d.data.year))})

.y0(d => yScale(d[0]))

.y1(d => yScale(d[1]))

.curve(d3.curveCatmullRom);

let stack = d3.stack()

.keys([...activeTabs])

.order(d3.stackOrderInsideout)

.offset(d3.stackOffsetWiggle);

layout.selectAll('path')

.data(stack(data))

.enter()

.append('path')

.style('fill', (d, i) => colors[activeTabs[i]])

.attr('d', area);

}

## 

## **2.4 Описание программы**

1) Общие сведения.

Для запуска подойдёт любая операционная система, поддерживающая работу браузера. Используемые css и js команды, поддерживаются всеми современными браузерами и internet explorer версии 8 и выше. Так же доступ к информации сайта можно получить через любое мобильное устройство, имеющее интернет.

2) Функциональное назначение – образовательный ресурс, содержащий информацию по некоторым вопросам здравоохранения.

3) Используемые технические средства.

ATOM – редактор текста.

D3.js – библиотека для работы с svg.

WebPack –сборщик проектов

koala – бесплатный компилятор препроцессора sass.

photoshop – для обработки изображений.

hover-master и magic-master – две css библиотеки, способствующие улучшению вида и интерактивности веб-сайта.

wireframe.cc ­ ресурс для построения шаблонов.

handpicked font ­ ресурс шрифтов.

ceaser ­ ресурс регулирующий плавность css анимаций.

free image online optimizer ­ ресурс для сокращения веса изображений.

can I use ­ последняя информация о совместимости инструментов и браузеров.

cdnsjs.com ­ поиск cdn библиотек.

Рекомендуемые *системные требования*: процессор Pentium­133 и выше, ОЗУ 16Мб, место на диске не меньше 4Мб.

## **2.5 Тестирование**

Для тестирования использовались ресурсы: cross browser testing tool и responsinator. Функциональная часть работы веб-сайта полностью исправна во всех браузерах и Internet Explorer версии выше 8. Так же веб-сайт одинаково хорошо работает на всех платформах: широкоформатный монитор 1920, обычный монитор 1380, ноутбук, планшет, смартфоны.

Для изучения работы сетевых процессов использовался набор встроенных инструментов для веб-разработчика браузера google chrome. Получена следующая информация: количество http запросов ­ 7, скорость загрузки страницы составила 1,08 сек.

# 3 Руководство оператора

После загрузки в окне браузера, веб­сайт представляет собой рабочий стол в центре которого расположен белый лист, он является рабочим пространством. Над ним расположен вводный заголовок, а под ним ссылка на источник данных (рис. 1):



#### Рисунок 1 – Окно приложения.

В нижней части рабочего пространства располагается панель управления, которая состоит из главного верхнего ряда и второстепенного нижнего.

Верхний ряд состоит из трёх пунктов, которые определяют форму представления информации, это гистограмма, поток и график. Нажатие на пункты верхнего ряда запрограммировано быть взаимозаменяемым.

Нижний ряд состоит из пяти пунктов, они представлены данными, которые будут визуализироваться. Пункты рисуют статистику врачей, медицинских работников, больничных организаций, амбулаторно-профилактических организаций, больничных коек. Пункты нижнего меню не являются взаимозаменяемыми для удобства сравнения статистики. Так же с этой целью они имеют цвет, совпадающий с цветом на рисунке (рис. 2, 3, 4).

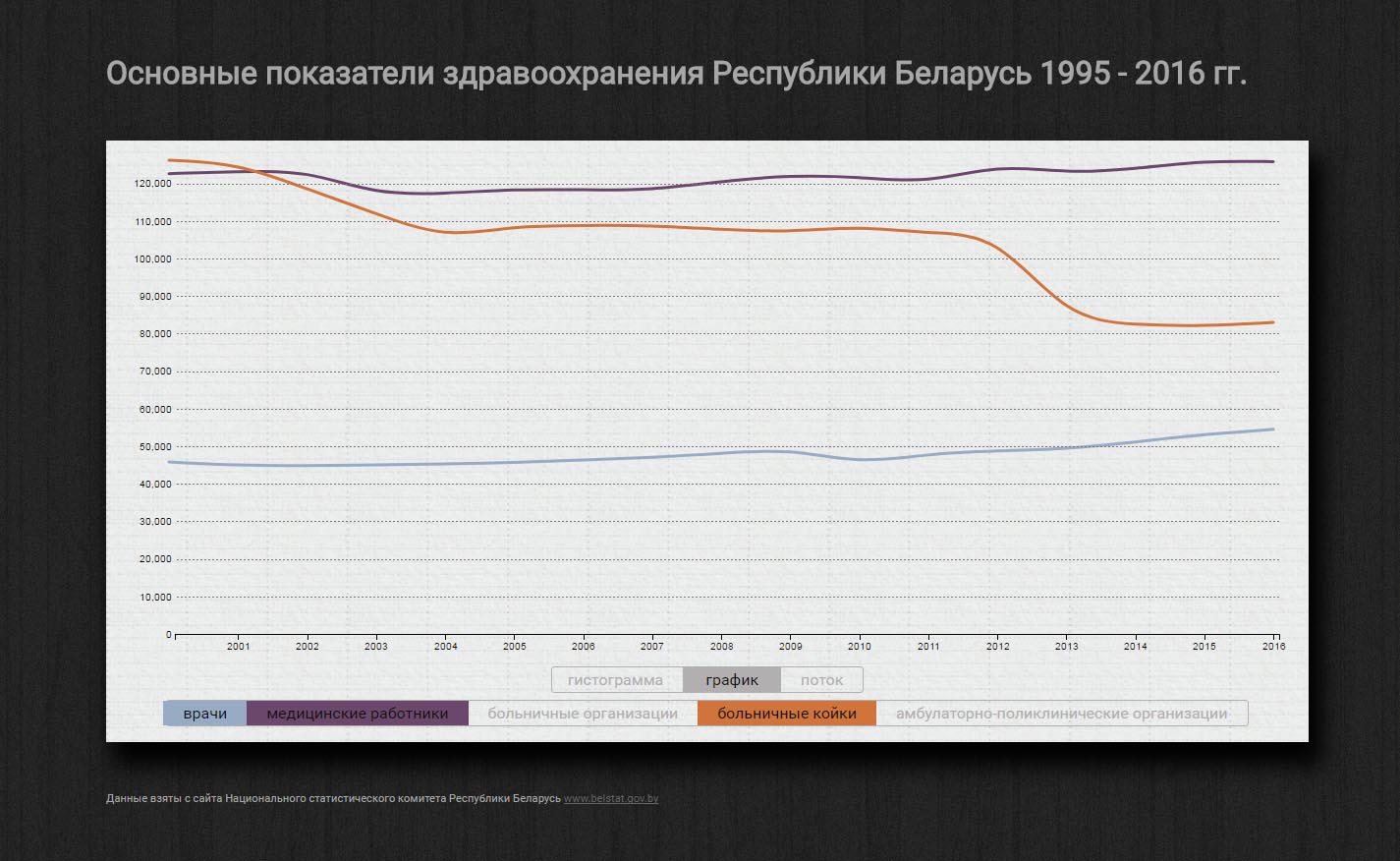
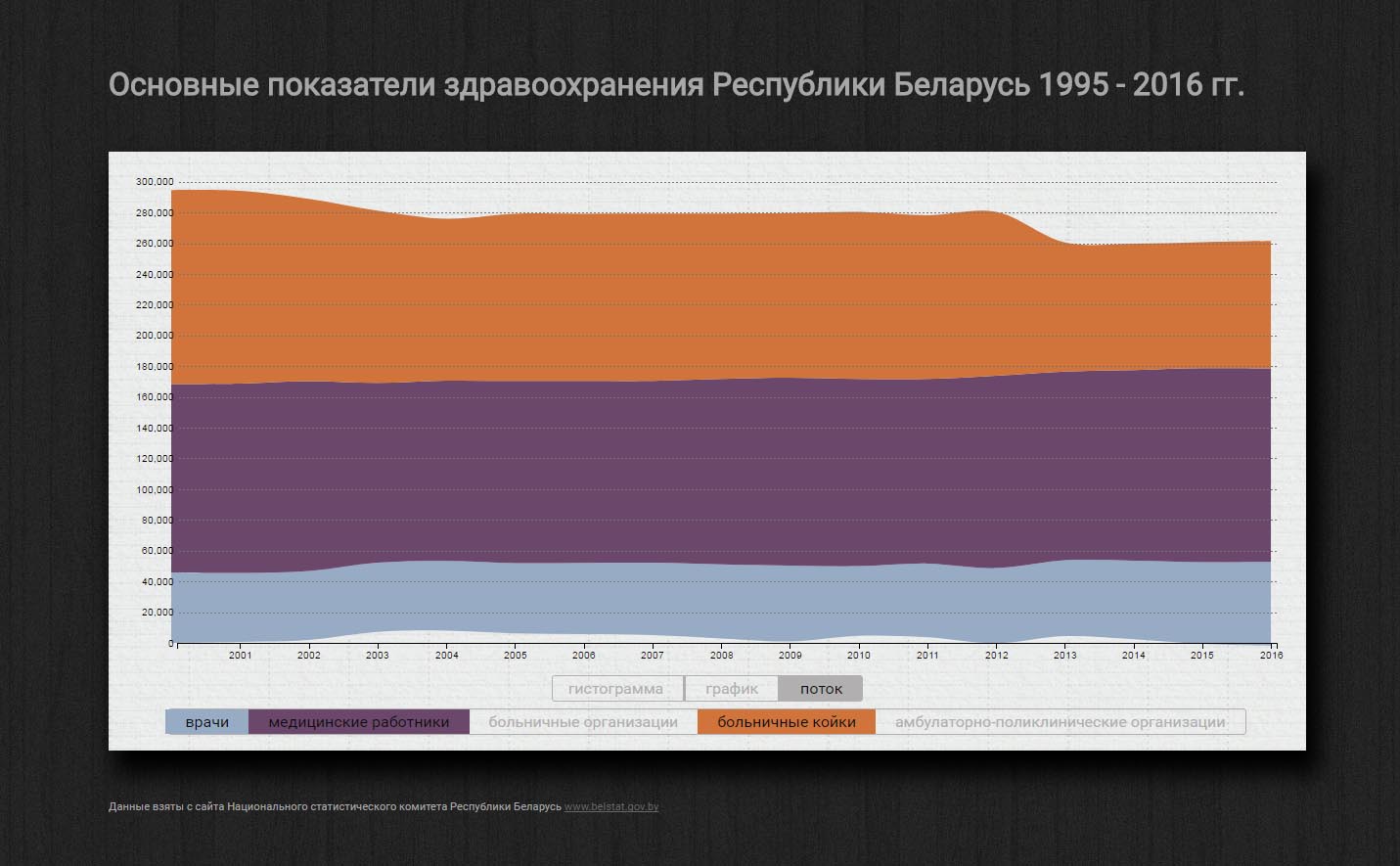


Рисунок 2 **­** окно приложения после выбора визуализации – «график».



#### Рисунок 3 **­** окно приложения после выбора визуализации – «поток».

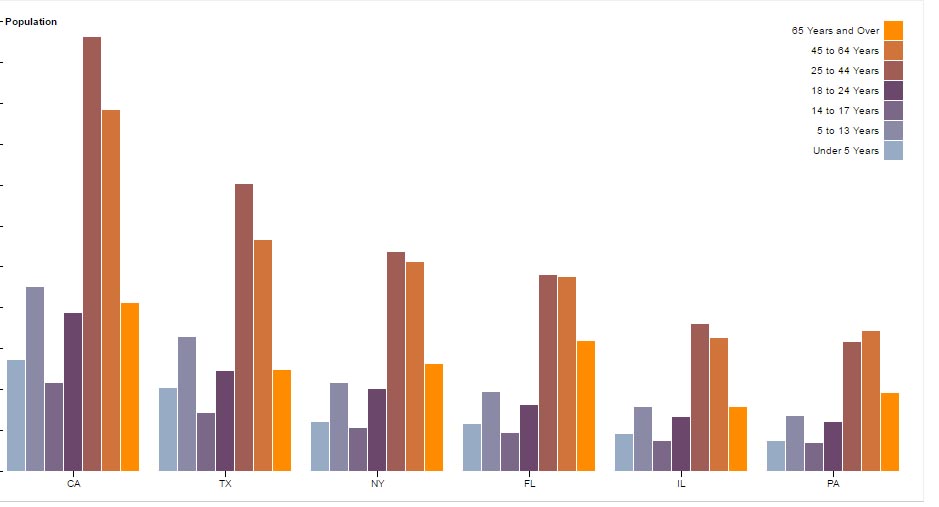


Рисунок 4 **­** окно приложения после выбора визуализации – «гистограмма».

Весь интерфейс является интуитивно понятным, не имеет лишних отяжеляющих элементов, увеличивающих когнитивную нагрузку, что делает ориентацию удобной для пользователей любого уровня.

# Заключение

Для образовательных целей разработан веб­сайт, содержащий информацию по специализированным вопросам здравоохранения. Раскрыты наиболее вероятные сценарии развития заболеваний, причиной которых является долгое и неправильное взаимодействие с компьютером. В ознакомительных целях даны описания заболеваний, их причины, симптомы, современные методы лечения и профилактика. Эта информация будет полезной людям, которые в силу различных причин, проводят значительное количество времени используя компьютер.

Веб­сайт имеет основную версию, корректно работающую на мониторах и ноутбуках всех разрешений, и две мобильные версии, для планшетов и смартфонов. Все версии веб-сайта поддерживаются браузерами всех типов, а так же internet explorer версии 8 и выше.

# Список использованных источников

1. Режим доступа: адрес ­ https://en.wikipedia.org/wiki/Information\_design. – Дата доступа: 5.05.2017.
2. Режим доступа: адрес ­ https://learn.javascript.ru/metrics-window. – Дата доступа: 5.05.2017.
3. Режим доступа: адрес ­ https://bl.ocks.org/mbostock/3887051. – Дата доступа: 6.05.2017.
4. Режим доступа: адрес ­ https://bl.ocks.org/mbostock/3312456 – Дата доступа: 6.05.2017.
5. Режим доступа: адрес ­ https://bl.ocks.org/mbostock/5433051. – Дата доступа: 6.05.2017.
6. Режим доступа: адрес ­ http://es5.javascript.ru/ – Дата доступа: 6.05.2017.
7. Режим доступа: адрес ­ https://www.ecma-international.org/publications/standards/Ecma-262.htm. – Дата доступа: 8.05.2017.
8. Режим доступа: адрес ­https://habrahabr.ru/post/252323/ – Дата доступа: 9.05.2017.
9. Режим доступа: адрес ­ https://habrahabr.ru/post/157087/ – Дата доступа: 9.05.2017.
10. Режим доступа: адрес ­ https://developer.mozilla.org/ru/docs/Web/SVG/Attribute – Дата доступа: 9.05.2017.
11. Режим доступа: адрес ­ https://d3js.org/ – Дата доступа: 9.05.2017.
12. Режим доступа: адрес ­ https://github.com/d3/d3/wiki/Gallery– Дата доступа: 9.05.2017.
13. Режим доступа: адрес ­ https://habrahabr.ru/company/datalaboratory/blog/217905/ – Дата доступа: 9.05.2017.
14. Режим доступа: адрес ­ https://metanit.com/web/d3js/1.1.php– Дата доступа: 10.05.2017.
15. Режим доступа: адрес ­ https://xakep.ru/2015/04/18/195-d3-js/– Дата доступа: 11.05.2017.
16. Режим доступа: адрес ­ https://www.dashingd3js.com/table-of-contents– Дата доступа: 11.05.2017.
17. Режим доступа: адрес ­ https://cdnjs.com/libraries/d3– Дата доступа: 12.05.2017.
18. Режим доступа: адрес ­ https://cdnjs.com/libraries/d3 – Дата доступа: 16.05.2017.
19. Режим доступа: адрес ­ http://bl.ocks.org/d3noob/b3ff6ae1c120eea654b5 – Дата доступа: 16.05.2017.
20. Режим доступа: адрес ­ https://bost.ocks.org/mike/bar/ – Дата доступа: 17.05.2017.