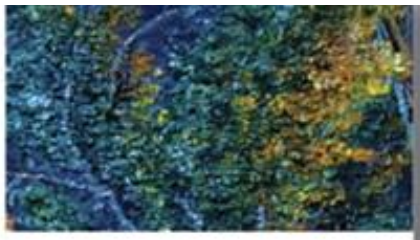

INTERNATIONAL SCIENCE REVIEWS

AIU

№ 2 (3) 2022

Natural Sciences and Technologies series

www.aiu.kz

ISSN: 2707-4862

ASTANA INTERNATIONAL UNIVERSITY

INTERNATIONAL SCIENCE REVIEWS
Natural Sciences and Technologies series

Has been published since 2020

№2 (3) 2022

Nur-Sultan

Editorial address: 8, Kabanbay Batyr avenue, of.316, Nur-Sultan,
Kazakhstan, 010000
Tel.: (7172) 24-18-52 (ext. 316)
E-mail: natural-sciences@aiu.kz

International Sciense Reviews NST - 76153
International Science Reviews
Natural Sciences and Technologies series
Owner: Astana International University
Periodicity: quarterly
Circulation: 500 copies

CONTENT

УДК 504.38

FAVOURABLE CLIMATIC CONDITIONS IN THE ALTAI MOUNTAIN COUNTRY

Ragulin V. S.

master student
Astana International University
Nur-Sultan, Kazakhstan
slava.ragulin@gmail.com

Annotation. Altai mountain country is an extremely attractive region for the development of tourism, both on the territory of Kazakhstan and on the territory of Russia, which only reinforces the relevance of this study. At the same time, the tourism industry is now becoming one of the fastest growing sectors of the economy.

Keywords: Altai mountain country, climate favorability indices, recreation, tourism

INTRODUCTION

Altai mountain country covers the territory of Kazakhstan, Russia, Mongolia and China, and is a unique natural object. The height of alpine ridges reaches more than 4000 m. The overall picturesque mountain massifs, the possibility of organizing different types of tourist routes, having a different category of difficulty (from the simplest 1 category to the most difficult 5 categories) [1], attracts tourists from around the world, especially in summer.

The complex mountainous terrain, which causes differences in solar exposure, air circulation and water balance, has a significant impact on the climate of the region.

The aim of this work is to study the favorability of the climate on the territory of the Altai mountain country.

METHODS AND SOURCES

A variety of climatic indices are used to assess the benefits: air temperature, humidity, wind speed, solar radiation, etc.

We selected the following as climate indices:

Effective Stationary Air Temperature (EN);

Equivalent Effective Temperature (EET);

Radiation Equivalent Effective Temperature (REET);

The severity index for Bodman.

Effective Temperature of Fixed Air (ET):

$$ET = t - 0{,}4(t - 10)\ (1 - f/100), \quad (1)$$

where f - relative air humidity; t - air temperature, °C.

The criteria in table 1 [2] are used to assess climate comfort under ET.

Table 1. Climate comfort criteria

Range ET, S°	Warmth	Load
more than 30	very hot (discomfort)	Strong
30-24	hot (warm subcomfort)	Moderate
24-18	warmth (comfortable-warm)	comfortably
18-12	moderately warm (comfortable-warm)	
12-6	Cool (cool subcomfort)	Moderate
6-0	moderately cool (cold discomfort)	
0 – -12	It's cold	
-12 – -24	very cold	strong threat of frostbite
-24 – -30	extremely cold	very strong frostbite threat
below -30		Extremely high probability of freezing

Equivalent Effective Temperature (EET)

EET - Thermal sensitivity with wind (2):

$$ЭЭТ = 37 - \frac{37 - t}{0,68 - 0,0014f + \frac{1}{1,76 + 1,4v^{0,75}}} - 0,29t(1 - \frac{f}{100})$$

where t- air temperature, oC; f- relative humidity, %; v- wind speed, m/s.

The EET describes the warmth of a person in the shadows, or the warmth of a dressed person. This indicator is well suited for the heat range, is satisfactory for the cooling range, and for the cold range it can be considered as an approximate criterion. For most people, the air temperature is 22.0-23.0°C and the relative humidity is 56%, which is close to 18° EET. Depending on the EET values, a cooling zone (1-17°C), a comfort zone (17-21°C) and a heating zone (above 21°C) [6].

Radiational Equivalent Effective Temperature (REET) [3]:

$$REET = 125lg[1 + 0,02t + 0,001(t - 8)(f - 60) - 0,45(33 - t)\sqrt{v} + 185B], \quad (3)$$

where t - air temperature, °C; f - relative air humidity, %; v - wind speed, m/s; B - surface absorbed solar radiation, kWt/m^2.

REET can also be calculated using the formula [4]:

$$REET = 0,83EET + 12 \quad (4)$$

REET is described as the most informative index:

discomfort: less 17°C;

subcomfort: 17-21°C;

comfort: 21–27°C;

subcomfort: 27-32°C;

discomfort: more 32°C.

Bodman severity index (S) [5]:

$$S = (1 - 0,04t)(1 + 0,27V), \quad (5)$$

where S - severity index (points), t - air temperature (°C), V - wind speed, m/s;

The following criteria are used to assess the severity of winter S:

S = 1 – winter is not harsh, mild;

S = 1-2 – winter is a little harsh;

S = 2-3 – moderately severe;

S = 3-4 – harsh;

S = 4-5 – very harsh;

S = 5-6 – brutally harsh;

S > 6 – extremely harsh.

In this study data of meteorological stations (MS) RGP «Kazgidromet» MEGIP RK averaged for the period 1991-2020 were used as primary data. for stations located on the territory of Kazakhstan and data of the weather archive from the Help and information portal «Weather and climate» [6-9].

RESULTS

Climatic assessment

On the territory of the studied region there are 16 meteorological monitoring stations that conducted observations for 10 years. of them, 3 are located on the territory of Kazakhstan (Shemonaikha station, Ust-Kamenogorsk station, Katon-Karagai station) and 13 on the territory of Russia (Charyshskoye station, Soloneshnoye station, Ust-Kan station, Gorno-Altaisk station, Kyzyl-Ozek station, Turochak station, Shebalino station, Chemal station, Ondugai station, Ust-Koksa station, Katanda station, Kara-Tyurek station, Ak-Kem station, Tashanta station) (fig. 1).

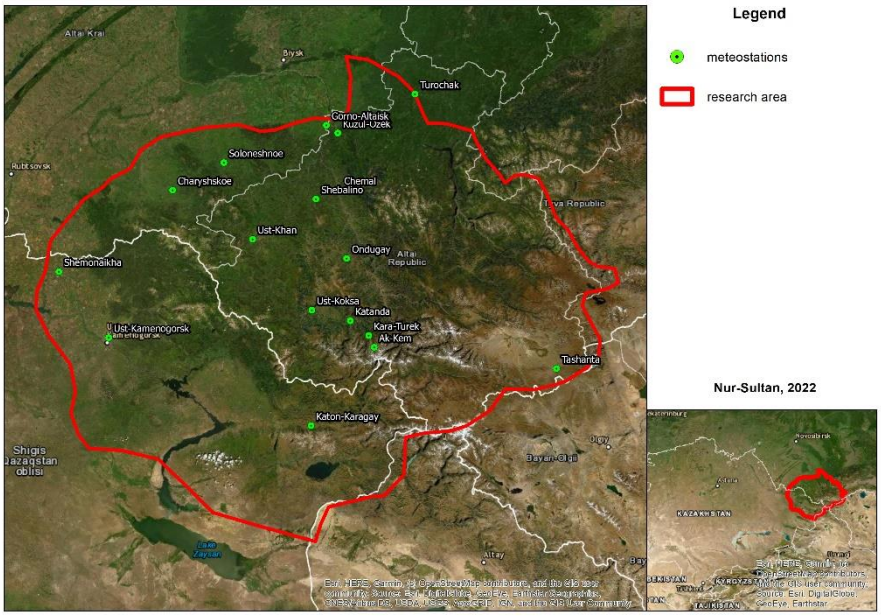

Figure 1. Location of meteorological stations

Monthly and annual averages of temperature and relative humidity as well as wind speeds for the period from 1991 to 2020 were taken into account for the calculation of the selected climate favorability indices.

Average monthly values of temperatures over a multi-year period at monitoring stations have a general pattern on average: the maximum falls in June-July (the largest value at the Ust-Kamenogorsk station is 20°C), at least in January (the smallest value at the Ust-Kan station is 32.18°C). Fig. 2 shows the annual air temperature at 7 stations, located on different relief exposures.

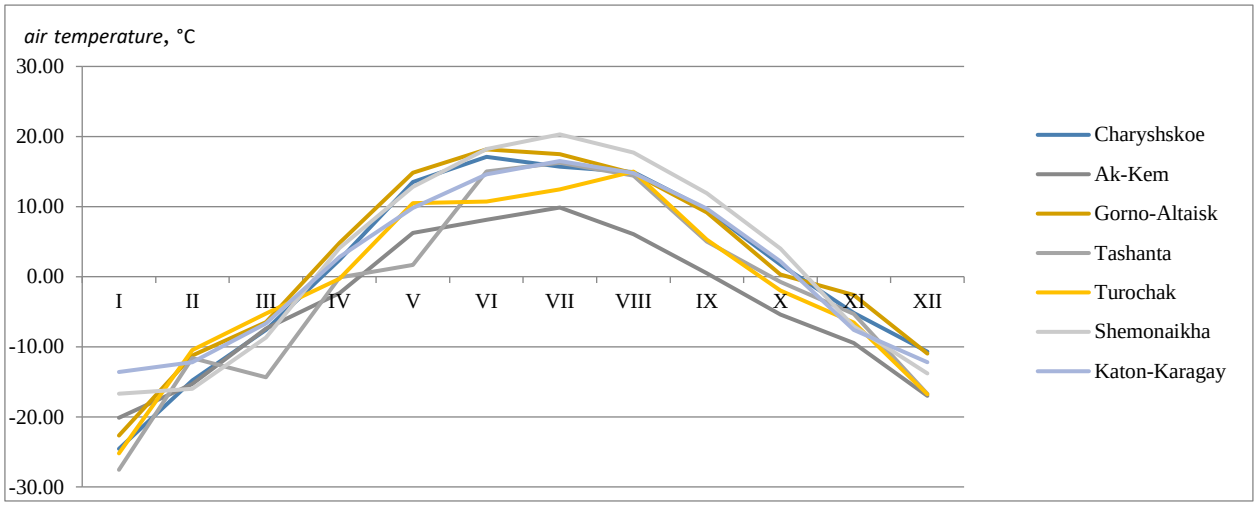

Figure 2. Annual course of air temperature

The relative humidity values are not consistent with the temperature values. However, a decrease in relative humidity at most weather stations in May can be discerned. This phenomenon can be explained by the complexity of the mountainous terrain, with a large number of hollows and west, accumulating moisture in shaded areas. The highest value (89.3%) was found at Charyshskoye station in March and Gorno-Altaisk station in October, the smallest (49%) - at Kara-Turkek station in February. Figure 3 shows the annual relative humidity of the 7 stations located at different relief exposures. The relative humidity of the 7 stations is measured in the same way.

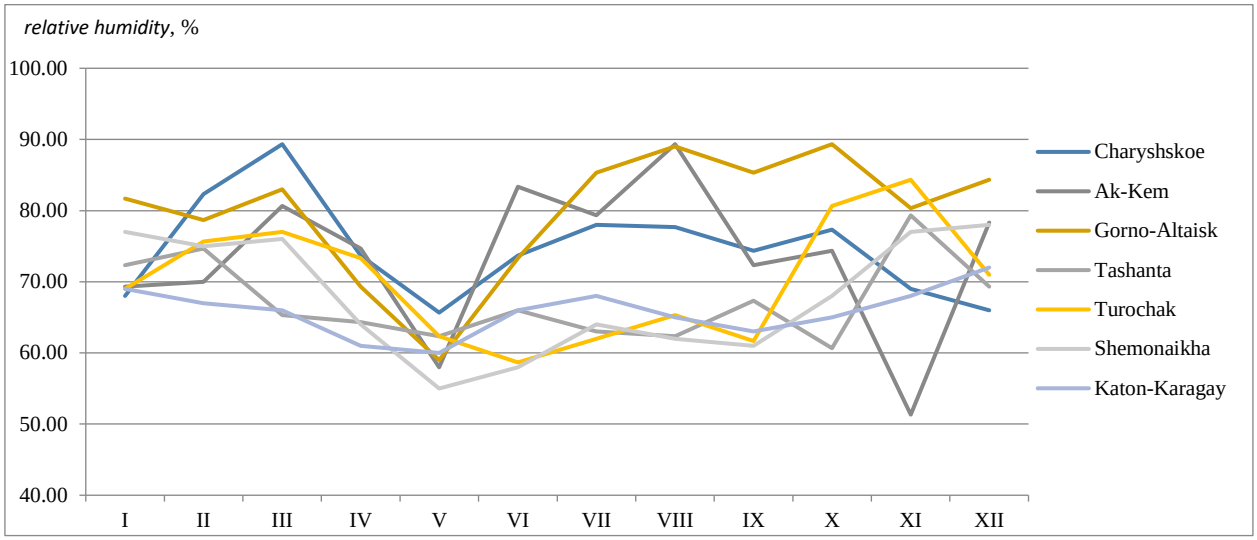

Figure 3. Annual stroke of relative humidity

The average monthly wind speed is not systematic. The maximum was found in March at the Katon-Karagai station and was 4.2 m/s, the minimum in January at the Ust-Kan station was 0.1 m/s. On figure. 4 shows the annual wind velocity of the air at 7 stations located on different relief expositions.

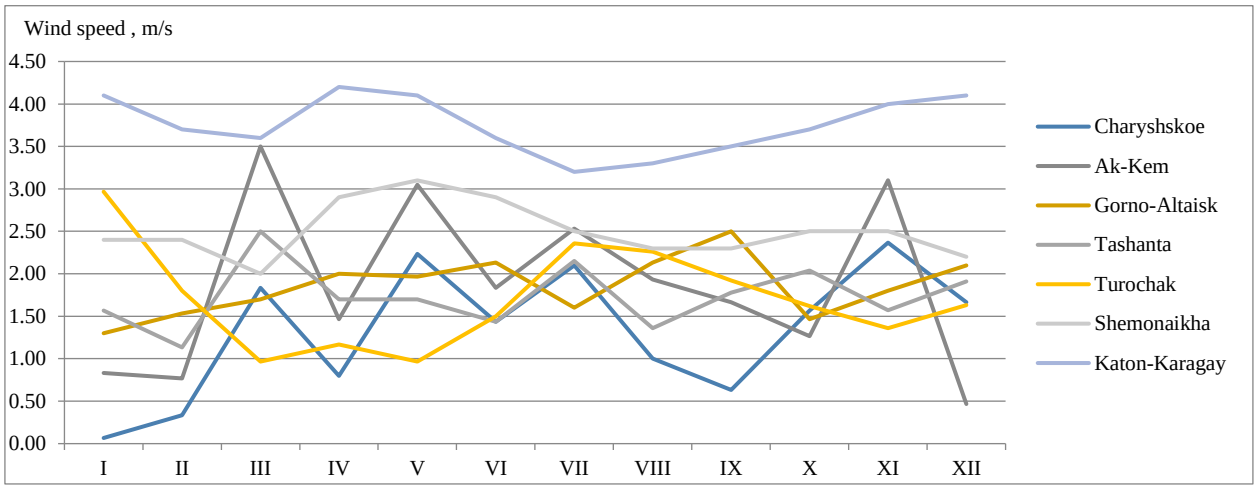

Figure 4. Annual course of wind speed

Climate benefit assessment

The effective temperature (ET) of the fixed air for each observation station was calculated using formula (1). The ET values at the stations were interpolated over the entire area of the study area using the ArcGIS Desktop software (Fig. 5).

According to the results obtained, it is possible to establish that the area with the most favorable climate, with moderate load and moderate-cool heat sensation is predominantly on the territory of Kazakhstan. The rest of the Territory is, to varying degrees, climatically cold, albeit moderately loaded. The most severe climate occurs in

the central part of the study area, which is due to the high mountain landscape, in particular the location of the highest point of the Altai Mountains - Belukha mountain.

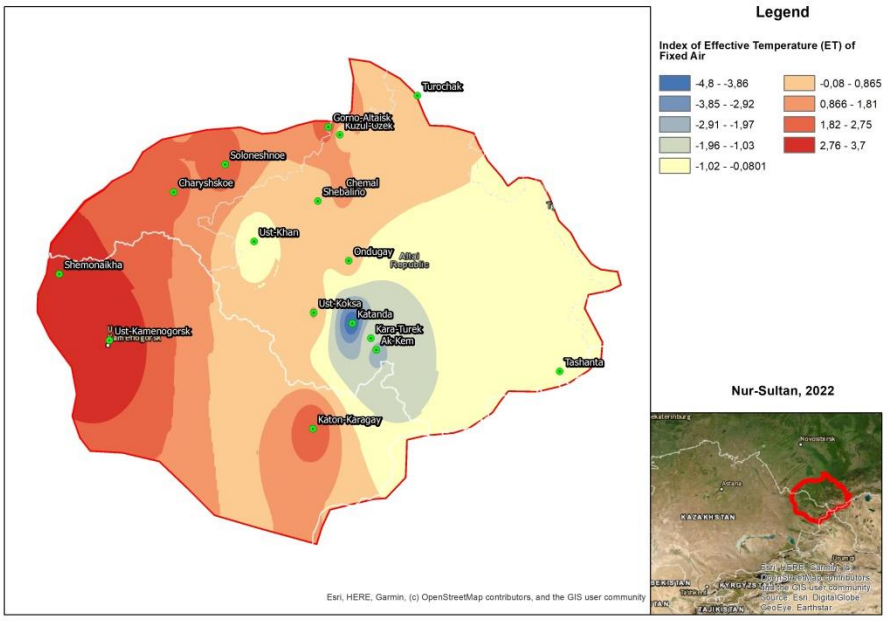

Figure 5. Spatial distribution of the effective air temperature

The equivalent effective air temperature (EET) for each station was then calculated using formula (2). EET values at the stations were interpolated over the entire study area using ArcGIS Desktop software (Figure 6).

According to Figure 6, the climate of the study area is unfavorable for humans. However, the zone with the relatively most favorable climatic conditions is concentrated mainly in the west and worsens to the east. Also, as in the calculation of ET, the lowest EET indicators prevail in the central part.

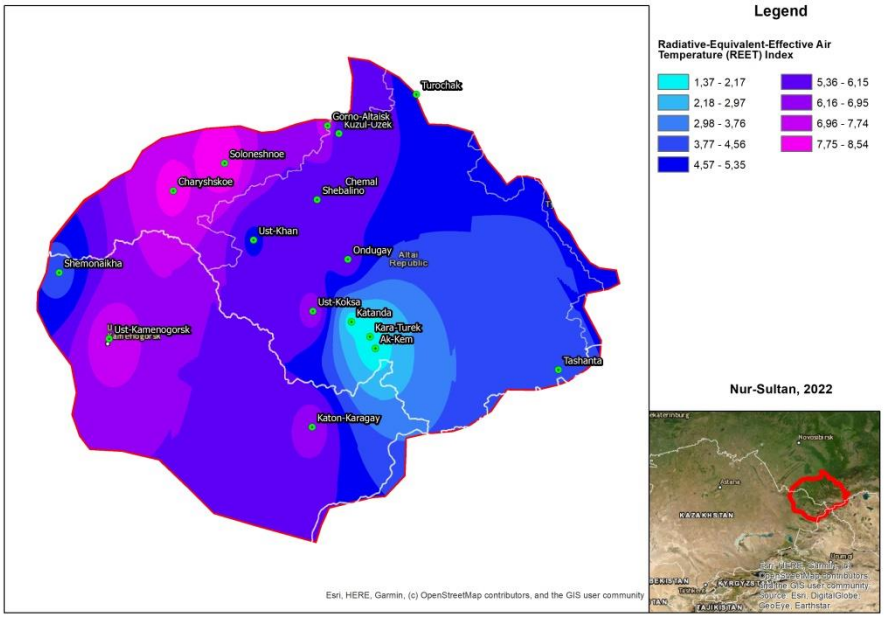

Figure 6. Spatial distribution of *equivalent-effective air temperature*

Then, using formula (4), the radiation-equivalent-effective air temperature (REET) for each station was calculated. The REET values at the stations were interpolated over the entire study area using ArcGIS Desktop software (Figure 7).

According to the results of the interpolation of REET values in the territory of the research area, the climate of the Altai Mountains is unfavorable for humans and belongs to the class - discomfort (less than 17 ° C).

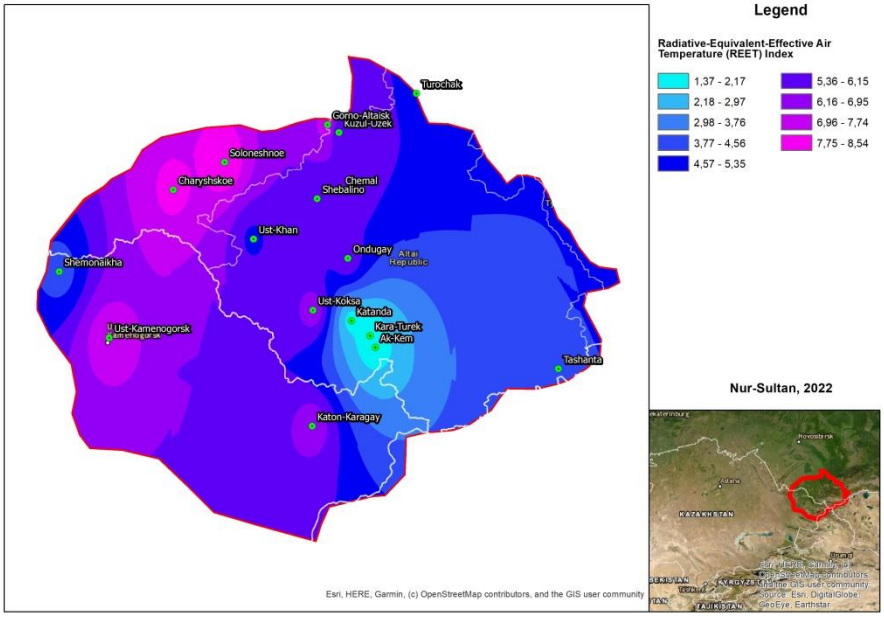

Figure 7. Spatial distribution of radiation-equivalent-effective air temperature

Using the formula (5) for the winter period (January-February, December), the Bodman severity index (S) was calculated (Fig. 8), according to which in the north-west of the study area the zone of a slightly severe winter prevails, in the area of the Katon-Karagai weather station there is a zone of severe winter, and the rest of the territory climatically refers to a moderately severe winter.

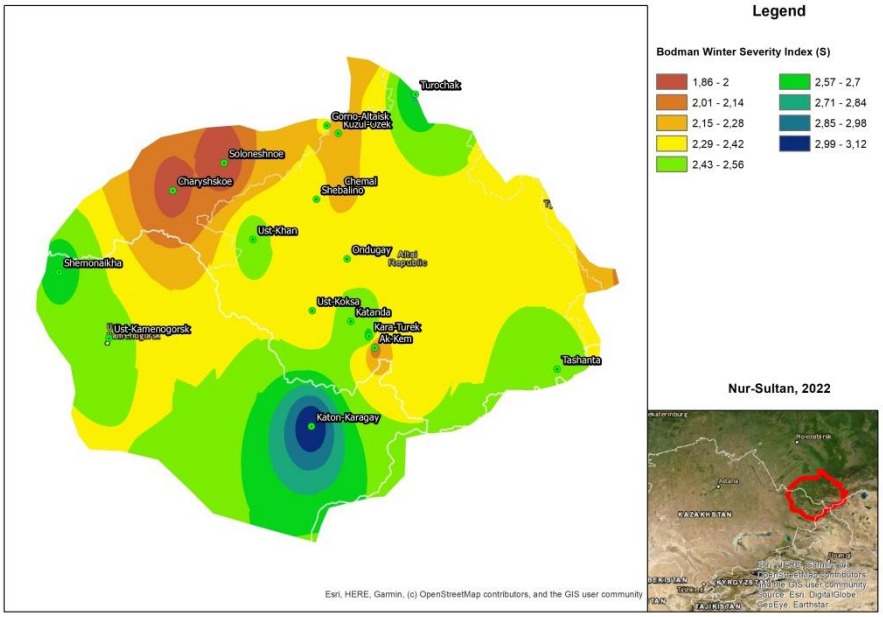

Figure 8. Spatial distribution of the *Bodman Winter Severity Index*

CONCLUSION

Within the study area, it is possible to distinguish a pattern of change of conditions from west to east for three average annual indices of climate favorability from more favorable to less favorable. It is also possible to distinguish the hotbed of the most severe (unfavourable) climatic environment in the central region of the study area, in particular at the foot of Mount Belukha.

In the cold period of the year, according to the Bodman Winter Severity Index, most of the area of the study area is moderately severe winter, except for the northwest (slightly severe winter) and the Katon-Karagai station (severe winter).

Thus, it can be concluded that the best type of tourism will be seasonal, as the average annual and winter climatic conditions are unfavourable to human health.

REFERENCES

1. Encyclopedia tourista, 1993. Pod red. Tamm E. I. Bolshaya rossiiskaya entsiklopediya. pp. 152--154

2. Bioclimatic indices in the assessment of the impact of modern climate warming on the living conditions of the population of Russia /V.V. Vinogradova // Izvestiya RAN. Sir. Geogr. 2009. - №3. S. 82 89.

3. Rusanov V.I. Complex meteorological indicators and methods of climate assessment for medical purposes / Tomsk: TGU, 1981. 86 s.

4. Golovina E.G. Some questions of biometeorology: ucheb. posobie /E.G.Golovina,V.I.Rusanov. SPb.: izd. RGMI, 1993. 90 s.

5. Isaev A.A. Ecological climatology. – M.: Nauch. mir, 2001. – 456 p.

6. "Handbook on climate of Kazakhstan. Section 1. Air temperature. Issue 10. East Kazakhstan Region" (RSE "Kazhydromet", Almaty, 2004);

7. "Handbook on climate of Kazakhstan. Section 10. Air Humidity" (RSE "Kazhydromet", Astana, 2013);

8. "Handbook on climate of Kazakhstan. Section 5. Wind. Section 6. Atmospheric pressure. Issue I-XIV" (RSE "Kazhydromet", Almaty, 2005);

9. Weather archive data from the Weather and Climate Reference and Information Portal http://www.pogodaiklimat.ru/ (accessed April 2022).

BOOTSTRAP. УПРОЩЕННАЯ РАЗРАБОТКА.

Кабиев Аслан, Нагим Шайдолла

Магистранты ВТиПО Международного университета Астана, г.Нур-Султан

Аннотация. ***Предмет.*** Bootstrap был разработан разработчиками Twitter в 2011 году для ускорения разработки и компоновки графических элементов интерфейса. В том же году команда выпустила его как фреймворк с открытым исходным кодом. На сегодняшний день Bootstrap является одной из наиболее часто используемых библиотек веб-разработки в мире.
Цели. Изучение Bootstrap. Поиск преимущества, а так же упрощенная разработка.
Методология. В процессе исследования использовались методы логического, статистического анализа.
Результаты. Разработка веб-приложений сводится к потреблению и публикации информации. Потребление, в частности, должно быть максимально легко усваиваемым. Это означает, что ваша информация должна быть доступна на любом устройстве. В визуально привлекательном и сбалансированном виде.
Выводы. Bootstrap прошел долгий путь с момента своего создания в качестве руководства по стилю для внутренних проектов Twitter. Теперь это зрелая, мощная интерфейсная библиотека, которая позволяет вам сосредоточиться на реализации вашего видения. В то же время он выполняет тяжелую работу по изменению размера вашего контента, чтобы он хорошо выглядел на экранах разных размеров.

Ключевые слова: bootstrap, веб-приложения, разработка, JS, CSS

Bootstrap — это мощная библиотека для разработки интерфейса, которая включает в себя готовые компоненты HTML , CSS и JS для создания веб-сайтов любого размера и сложности, ориентированных на мобильные устройства. С Bootstrap вы готовы использовать адаптивные темы и шаблоны в качестве отправной точки для своего веб-проекта.

Bootstrap включает шаблоны на основе HTML и CSS для типографики, форм, кнопок, таблиц, навигации, модальных окон, каруселей изображений и многих других, а также дополнительные плагины JavaScript. И хотя он не реализует шаблон MVC (модель-представление-контроллер), он придерживается разделения задач. Он также используется для создания представлений в шаблонах MVC, используемых в средах с полным стеком, таких как Django и ASP.NET MVC среди прочих.

Зачем использовать Bootstrap?

Разработка веб-приложений сводится к потреблению и публикации информации. Потребление, в частности, должно быть максимально легко усваиваемым. Это означает, что ваша информация должна быть доступна на любом устройстве. В визуально привлекательном и сбалансированном виде.

Как использовать бутстрап

Bootstrap очень прост в использовании: любой, у кого есть базовые знания HTML и CSS, может начать использовать Bootstrap (рисунок 1).

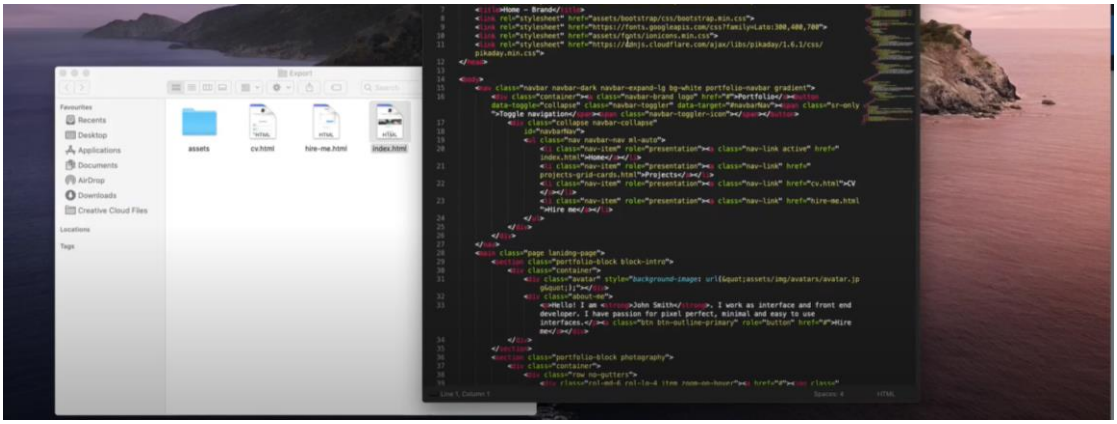

Рисунок 1. Окно Bootstrap

Он имеет адаптивные функции:

Отзывчивый CSS Bootstrap адаптируется к устройствам всех размеров и разрешений.

Совместимость с браузерами: он совместим со всеми современными браузерами (Opera, Chrome, Firefox, Internet Explorer (Edge) и Safari).

В Интернете доступны сотни превосходно сконструированных бесплатных и монетизированных тем, которые дадут вам преимущество в работе. Наличие базового дизайна для работы действительно ускоряет разработку пользовательского интерфейса.

Важно отметить, что на официальном сайте Bootstrap есть отличная документация, которая поможет вам начать работу.

Вы можете загрузить его и разместить вместе с другими файлами приложений на своем веб-сервере или добавить ссылку на сеть доставки контента (CDN), которая всегда будет предоставлять вам самую последнюю версию.

Что думают разработчики?

Хотя популярность Bootstrap, возможно, упала за последние несколько лет, он по-прежнему широко используется. Мы попросили одного из наших постоянных разработчиков, Мэтта Раджа, рассказать нам о его взглядах на Bootstrap.

AKA План

Потом, когда же, появился CSS, который все одновременно упростил и усложнил. Теперь вещи можно было позиционировать с точностью до пикселя, а качество веб-разработки быстро улучшилось. Многие разрабочики по-прежнему предпочитают писать CSS с нуля. Однако может показаться, что вы играете в особенно напряженную игру, где один неверный ход может разрушить все. Одним из решений бесконечных настроек CSS является использование интерфейсной среды или библиотеки, где вся тяжелая работа уже сделана за вас. Bootstrap — это то, к чему я всегда стремлюсь.

Bootstrap начал свою жизнь как проект Twitter еще в 2011 году. Первоначально он назывался «Blueprint», имя, которое я предпочитаю! Марк Отто, разработчик, инициировавший проект, сказал: «Мы с очень небольшой группой разработчиков собрались вместе, чтобы спроектировать и создать новый внутренний инструмент, и увидели возможность сделать что-то большее. В ходе этого процесса мы увидели, что создали нечто гораздо более существенное, чем еще один внутренний инструмент. Спустя несколько месяцев у нас появилась ранняя версия Bootstrap, которая позволяла документировать и делиться общими шаблонами проектирования и активами внутри компании».

Вот пять причин, по которым Bootstrap еще актуален:

Сверхбыстрая разработка. Простая для понимания и гибкая сеточная система, а также обширная библиотека подключаемых компонентов позволяют создать функциональный и адаптивный веб-сайт за короткий промежуток времени. Элементы легко появляются или исчезают в зависимости от размера экрана. Bootstrap основан на концепции разработки для мобильных устройств, поэтому ваш контент будет идеально масштабироваться для мобильного устройства без необходимости делать слишком много. Затем вы можете решить, какие компоненты должны быть видны или как содержимое должно корректироваться при изменении размера экрана. Прямо из коробки вы можете очень быстро запустить проект. Это, конечно, означает, что у вас есть обычный сайт на Bootstrap, но это подводит нас ко второй причине, почему мне нравится Bootstrap.

Его легко настроить. С небольшим знанием CSS все визуальные элементы в Bootstrap можно настраивать. Это означает, что обычный проект Bootstrap не должен оставаться таким. Вы можете изменить дизайн кнопок, модальных окон,

панелей навигации, ссылок и меню, и многие крупные организации делают это, чтобы персонализировать взаимодействие с пользователем. Веб-сайт Forbes India (ссылка: forbesindia.com) использует ванильный Bootstrap. Даже на основном веб-сайте Forbes (ссылка: forbes.com) используется модифицированная версия Bootstrap 3. Таким образом, если немного поработать, проект Bootstrap не обязательно должен выглядеть как проект Bootstrap, что прекрасно переходит в мою третью причину:

Вам не обязательно использовать все это приверженность использованию Bootstrap не означает, что вам нужно использовать все это. Если вам не нравятся кнопки Bootstrap по умолчанию, создайте свои собственные в CSS и оставьте Bootstrap для супер-простой отзывчивой сетки. Вы можете легко создавать части вашего проекта по индивидуальному заказу и добавлять Bootstrap там, где это необходимо.

Он имеет массивную библиотеку интерактивных компонентов Складные мобильные меню? Интерактивные слайдеры-карусели? Стилизованные всплывающие окна? Прокрутите-шпион, когда вы спускаетесь вниз по одностраничному приложению? Поповеры и спиннеры? Все они включены в Bootstrap. Теперь некоторые из них, на мой взгляд, используются слишком часто. Я мог бы разглагольствовать о карусели дольше, чем вам хотелось бы продолжать чтение; однако при правильном и разумном использовании вы можете создавать красиво оформленные интерактивные компоненты на своем сайте практически без программирования. Большинство из них просто требуют добавления простого имени класса к элементу . Все эти компоненты также настраиваются, поэтому вы можете вписать их в общую тему вашего сайта. И, наконец, если вы застряли при настройке этих компонентов:

Существует значительное сообщество, активно развивающееся для Bootstrap. Существуют тысячи шаблонов и примеров того, как вы можете настроить Bootstrap. Взгляните на примеры на CodePen , чтобы увидеть творческие способы настройки Bootstrap и — с правильным указанием авторства, конечно — попробуйте некоторые из них в своих проектах. С небольшой помощью сообщества вы можете вывести свой проект Bootstrap на новый уровень.

В заключении, Bootstrap прошел долгий путь с момента своего создания в качестве руководства по стилю для внутренних проектов Twitter. Теперь это зрелая, мощная интерфейсная библиотека, которая позволяет вам сосредоточиться на реализации вашего видения. В то же время он выполняет тяжелую работу по изменению размера вашего контента, чтобы он хорошо выглядел на экранах разных размеров.

СПИСОК ЛИТЕРАТУРЫ

1. Ф.А. Перепелица Эффективная разработка веб-сайтов. Bootstrap Учебное пособие СПб, 2017. С. 247-273.

2. ИТ Блог. Администрирование серверов на основе Linux (Ubuntu, Debian, CentOS, openSUSE) // Учебное пособие по framework Bootstrap [Электронныйресурс]: исследование, проведенное PwC. – М. – 2019. – Режим доступа: https://andreyex.ru/uchebnoe-posobie-po-framework-bootstrap/ (дата обращения 20.07.2020).

3. Исследование, проведенное Copyright © 2020 Rudebox Design [Электронный ресурс] – Режим доступа: https://www.rudebox.org.ua/bootstrap-5-what-is-new-and-release-date. (дата обращения 21.07.2020).

4. Обновление Bootstrap 5 и генератор статических сайтов Hugo /. Web разработка [Электронный ресурс] – Режим доступа: http://tods-blog.com.ua/web-development/bootstrap5-hugo.

ВЕБ-ФРАЙМВОРК ANGULAR. ТЕСТИРОВАНИЕ. БИБЛИОТЕКА REDUX

Кабиев Аслан, Нагим Шайдолла

Магистранты ВТиПО Международного университета Астана, г.Нур-Султан

Аннотация. ***Предмет.*** Angular – это фреймворк с открытым исходным кодом, поддерживаемый Google. Впервые разработанный в 2010 году как AngularJS, фреймворк обновлялся почти каждый год по архитектуре, синтаксису и производительности, чтобы работать без сбоев со всей экосистемой JavaScript.
Цели. Изучение веб-фреймворка Angular, тестирование. А так же расширенное изучение библиотеки Redux.
Методолгия. В исследовании использовались методы логического, статистического анализа.
Результаты. **Тестирование в Ангуляре**
Тот факт, что компоненты независимы друг от друга, значительно упрощает модульное тестирование. Действительно, когда проект Angular настраивается с использованием angular-cli, он поставляется с Jasmine+Karma, все 14 настроены для разработчика. Тем не менее, у разработчиков есть возможность использовать различные средства запуска тестов, такие как Jest.
Выводы. Сделан вывод о том, что Ангуляр хорош для создания single-page applications, а создание виджетов, которые можно использовать на любой HTML-странице, - непростая задача. Новый модуль Angular Elements позволит создавать отдельный компонент и публиковать его как Web Component, который можно будет использовать на любой HTML-странице.

Ключевые слова: Angular, фреймворк, библиотека, Redux, функция

Во-первых, давайте разберемся, что такое одностраничные приложения (также известные как SPA), о которых все говорят!

Одностраничные приложения — это веб-приложения, которые работают в браузере и не требуют перезагрузки страниц в течение всего срока службы приложения, а туда и обратно передаются только данные.

Давайте углубимся в подробности: Angular — это фреймворк с открытым исходным кодом для фронтенд-разработки, который облегчает создание сложных, эффективных и сложных одностраничных приложений.

На самом деле, Angular v2+ был разработан разработчиками Google еще в 2012 году, чтобы убрать ненужный код и, как следствие, сделать приложение более легким и быстрым. Он основан на модулях, что делает его структуру более понятной и удобной в сопровождении. Это также упрощает разделение рабочей

нагрузки между разработчиками в одной команде, что повышает производительность и сотрудничество между ними, тем самым улучшая качество кода.

Кроме того, Angular поставляется с полным набором полезных инструментов, таких как собственный CLI, утилиты отладки, материал Angular, ng Bootstrap и т. д. Например, CLI экономит огромное количество времени для разработчиков благодаря своей способности настроить проект Angular. это довольно сложный и очень долгий процесс, если он выполняется вручную с нуля, поскольку необходимо установить так много конфигураций, так много библиотек и файлов. Он также создает компоненты, директивы, защиту, сервисы и т. д. и добавляет весь шаблонный код, идущий с каждым из них. С другой стороны, он создает приложение для запуска в браузере или для развертывания. Angular CLI также используется для тестирования приложения.

Кроме того, приложения Angular используют Typescript, который является надстрочным индексом JavaScript. Фактически, Typescript упрощает код JavaScript, облегчает его чтение и отладку. Он также обеспечивает более высокую безопасность, поскольку поддерживает типы, классы, интерфейсы и устраняет ошибки на ранней стадии при написании кода. Кроме того, машинописный код можно отлаживать непосредственно в браузере или в редакторе, что обеспечивает услуги рефакторинга и автоматического завершения. Более того, функции получения и установки не нужны при использовании Angular, поскольку он предоставляет все функции, предоставляемые JavaScript, а это означает, что объекты, используемые Angular, являются POJO (обычный старый объект JavaScript).

Очень важно подчеркнуть, что Angular — это архитектура, основанная на компонентах, которая обеспечивает согласованность кода, делает упор на повторное использование кода, упрощает модульное тестирование, поскольку компоненты являются независимыми модулями. По сути, все компоненты, сервисы, директивы, пайпы и т. д. созданы в едином стиле. В качестве примера давайте декортикируем, как выглядит компонент. Каждый компонент определяет декоратор, который является @Component, в котором он указан: Селектор: будет использоваться из других компонентов для его вызова. Template/templateUrl: содержит HTML-представление. Может быть встроенным или в отдельном файле. Style/StyleUrls: содержит стиль для шаблона. Может быть встроенным или в отдельном файле. И весь код помещается в класс компонентов.

Структура углового компонента

Угловые шаблоны проектирования:

Angular следует различным шаблонам проектирования, некоторые из них представляют собой реактивное программирование, централизованный поток управления состоянием и однонаправленные данные, которые стоит обсудить.

Реактивное программирование:

Angular использует RxJS (библиотека реактивных расширений для JavaScript) для реализации шаблона Observable, который обрабатывает загрузку данных асинхронно, что делает современное веб-приложение намного лучше по сравнению со старыми приложениями, которые раньше останавливались в разных точках, что усложняло работу пользователя.

Паттерн Observable можно объяснить с помощью следующей диаграммы:

Наблюдаемый шаблон проектирования

Централизованное управление состоянием:

Основная цель централизации состояния в приложении — сделать изменяющиеся данные управляемыми, поддерживаемыми и предсказуемыми, что достигается за счет соблюдения некоторых ограничений, таких как знание того, как, где и когда могут происходить обновления с использованием однонаправленного потока данных.

Преимущества этого шаблона проявляются в больших приложениях, где запоминание состояния различных компонентов и потоков данных может оказаться отвратительной задачей, а в некоторых случаях беспорядочной и неконтролируемой.

Angular использует Redux для управления централизованным состоянием. Redux — это и библиотека, и шаблон проектирования, основанный на следующих трех принципах:

Единственный источник правды:

Это означает, что состояние всего приложения представляет собой единую неизменяемую структуру данных, хранящуюся в дереве объектов в одном хранилище данных.

Состояние доступно только для чтения:

Это означает, что состояние никогда не изменяется напрямую, вместо этого изменения инициируются действием, описывающим, что произошло.

Изменения вносятся с помощью чистых функций:

Изменения состояния вносятся в чистые функции, называемые редьюсерами, которые принимают предыдущее состояние и следующее действие для создания нового состояния.

Однонаправленный поток данных:

Однонаправленный поток данных — это шаблон управления данными, применяемый Angular для упрощения реакции приложений на изменения. Префикс uni определяет, что данные могут передаваться только в одном направлении. У Angular есть полный жизненный цикл создания, компоновки, проверки и уничтожения представления всякий раз, когда представление отображается. Другими словами, это означает, что каждый раз, когда что-то меняется, представление перерисовывается. С другой стороны, сохраняя однонаправленный поток данных, Angular обеспечивает двустороннюю привязку, что буквально означает две односторонние привязки для привязки свойств и событий.

Кроме того, команда Angular предоставляет полную документацию о том, как создать приложение с хорошей архитектурой, что упрощает новичкам выполнение основных требований для получения чистого и поддерживаемого кода.

СПИСОК ЛИТЕРАТУРЫ

1. Моисеев А., Яков Ф. Angular и TypeScript. Сайтостроение для профессионалов. 2017

2. Пабло Дилеман Изучаем Angular 2. 29 января 2022

3. Джереми Вилкен Angular in Action. 2018

4. Павел К., Питер Б., Разработка веб-приложений с использованием AngularJS. 29 января 2022.

ПРЕИМУЩЕСТВА КЛАССИЧЕСКИХ АРХИТЕКТУРНЫХ РЕШЕНИЙ ДЛЯ ВЕБ-ПРИЛОЖЕНИЙ.

Кабиев Аслан, Нагим Шайдолла

Магистранты 2 курса Международного университета Астана, г.Нур-Султан

Аннотация. ***Предмет.*** Современное веб-приложение, очевидно, реализовано на JavaScript и будет генерировать свой HTML-код на клиенте в браузере. Он взаимодействует с сервером только путем получения данных в формате JSON из конечной точки HTTP/REST API — это, кажется, общепринятое мнение на сегодняшний день. Но действительно ли серверный HTML и прогрессивное улучшение устарели? Наоборот, мы можем использовать эти технологии для создания приложений, которые часто намного лучше, чем те, которые мы могли бы создать с помощью Framework of the Week для создания одностраничных приложений.

Цели. Изучение преимущества классических архитектурных решений для веб-приложений.

Методология. В процессе иследования использовались методы логического, статического анализа

Результаты. На практике мы убедились, что когда мы разрабатываем приложения с серверным рендерингом HTML и пользовательскими элементами, тщательно модульным CSS и небольшим количеством JavaScript, мы можем создавать архитектурно чистые, компактные, быстрые и эргономичные приложения. Мы не боимся сравнения этих приложений с приложениями, созданными с использованием популярных подходов SPA — во многих аспектах они часто превосходят их. Мы обнаружили, что они не только имеют меньшие коммуникационные издержки, более высокую отказоустойчивость и более легко доступны, но и их легче поддерживать в долгосрочной перспективе.

Выводы. Мы зависим от браузера и его стандартов, а не от фреймворка SPA, который может полностью измениться в следующей версии или может быть заменен следующей альтернативной фреймворком.

Ключевые слова: архитектура, веб-приложения, CSS, HTML, сервер

Давайте рассмотрим проблемы, которые часто возникают при использовании одностраничных приложений (SPA). При традиционном разделении между сервером и клиентом клиент отвечает за внешний вид и поведение. Состояние, бизнес-логика, маршрутизация, логика представления и шаблоны находятся исключительно на сервере (рис. 1).

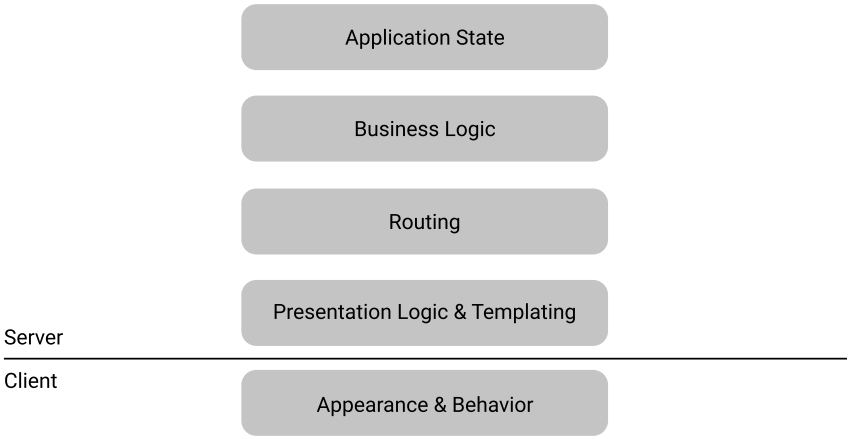

Рисунок 1. Традиционное разделение между сервером и клиентом

Когда мы создаем приложение и хотим оптимизировать его, чтобы оно быстрее реагировало и, возможно, предоставляло автономные возможности, мы часто переносим некоторые обязанности с сервера на клиента. Многие компании считают дополнительным преимуществом, если мы можем разделить эти обязанности между разными отделами. Тогда экспертам по бэкенду нужно только предоставить API, но им не нужно заботиться о HTML/CSS/JS. Кроме того, существует распространенное заблуждение, что JSON меньше, чем HTML. Джон Мур объясняет в статье , почему это не так. Часто логика представления и шаблоны перемещаются во внешний интерфейс. Это относится к таким технологиям, как React или Vue.js (рис. 2).

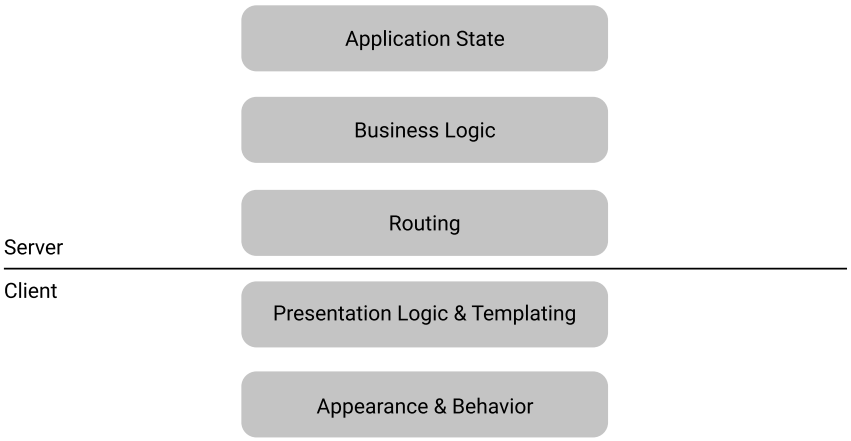

Рисунок 2. Логика представления и шаблоны перенесены во внешний интерфейс

Следующим шагом часто является передача маршрутизации клиенту. Например, это происходит в React с React Router и в Vue.js с Vue Router. В таких фреймворках, как Angular или Ember, клиент также отвечает за маршрутизацию (рис. 3).

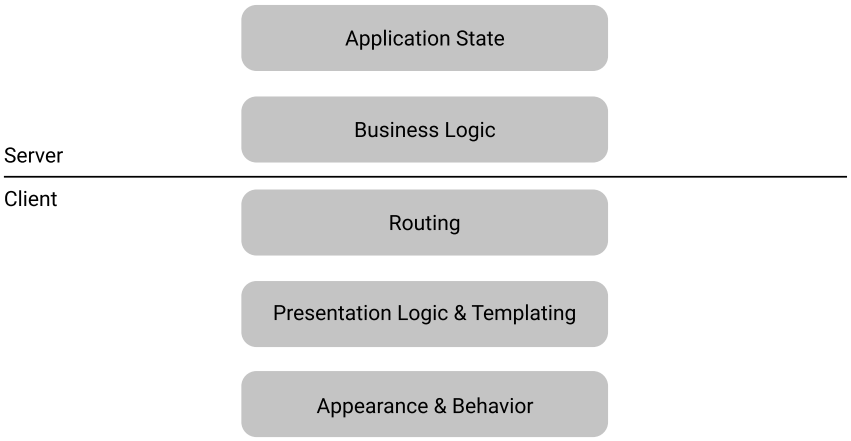

Рисунок 3. Маршрутизация перемещается в клиент

Может показаться, что перекладывание обязанностей с сервера на клиента — это всего лишь смена границ. Однако это изменение границы имеет дополнительные последствия, которые нам необходимо рассмотреть.

Раньше мы только интерпретировали HTML в клиенте — функциональность, которую браузер предоставляет из коробки. Теперь нам нужен клиент JSON и API, которые мы можем использовать. Это также добавляет уровень косвенности: в логике представления у нас больше нет прямого доступа к бизнес-логике и состоянию приложения, но вместо этого мы должны преобразовывать данные в JSON и обратно. Определение и изменение этого API требует высокого уровня координации и взаимодействия, особенно когда сервер и клиент разрабатываются разными командами.

Если и бизнес-логика, и состояние приложения полностью остаются на сервере, у нас либо такое же, либо больше взаимодействие между клиентом и сервером. Однако если мы переместим часть бизнес-логики на клиент, нам теперь нужно будет постоянно решать, какая бизнес-логика принадлежит серверу, какая бизнес-логике принадлежит клиенту, а какая должна быть продублирована в обоих. Здесь нужно обратить внимание на то, что клиент не является безопасной или доверенной средой. По этой причине нам также нужно проверить много вещей на сервере. GraphQL — это пример технологии, которая пытается использовать серверную часть только как «базу данных» без какой-либо бизнес-логики на сервере.

Состояние приложения также больше не находится только на сервере (база данных/сеанс…). Мы также должны поддерживать состояние в клиенте. Мы можем добиться этого с помощью таких решений, как LocalStorage/PouchBD, а также с помощью таких технологий, как Redux, Flux, Reflux и Vuex. Здесь нам также нужно решить, какие данные принадлежат клиенту, какие данные принадлежат серверу, а какие нужно сохранить в обоих местах. Нам также необходимо подумать о (многолидерной) синхронизации: поскольку данные могут быть одновременно изменены более чем на одном клиенте, который может быть отключен в течение более длительного периода времени, обязательно возникнут конфликты записи, которые необходимо разрешить.

Изменение границы между клиентом и сервером также означает, что нам нужно выполнять больше кода JavaScript на клиенте. Время, необходимое браузеру для разбора и выполнения кода JavaScript, многими недооценивается. Это также означает, что страницы с большим количеством JavaScript могут иметь более медленное время запуска. Чтобы избежать этого, многие фреймворки предлагают гидратацию: приложение сначала будет отображаться на сервере с помощью Node и доставляться клиенту в виде HTML. Когда JS загружается, он берет на себя разметку, и с этого момента приложение отображается на клиенте.

Еще одной причиной гидратации является SEO: поскольку JS не может надежно интерпретироваться поисковыми системами, имеет смысл предоставлять контент в виде HTML. Благодаря гидратации мы добавляем не только дополнительную сложность клиенту, но и дополнительную инфраструктуру.

Исполнение в неконтролируемой среде

Браузеры также являются более неконтролируемой средой выполнения, чем наш сервер. Нам нужно обратить на это внимание, когда мы загружаем больше кода в браузер. Один момент заключается в том, что наблюдаемость хуже. На сервере мы можем вести журналы и ловить и регистрировать исключения. Мы также можем сделать это на клиенте, но это намного сложнее и легче ошибиться. Мы также должны обратить внимание на некоторые факторы, касающиеся производительности. Мы часто тестируем наш JavaScript на очень быстрых машинах. Но на мобильных устройствах время, необходимое для обработки JavaScript, очень велико. Подробности можно найти здесь и здесь .

Браузеры также являются более неконтролируемой средой выполнения, чем наш сервер. Нам нужно обратить на это внимание, когда мы загружаем больше кода в браузер.

JavaScript выполняется в одном потоке. Только операции ввода-вывода выполняются асинхронно вне этого потока. Задача ЦП блокирует поток. Таким образом, задачи с интенсивным использованием ЦП приводят к тому, что страница

перестает отвечать на запросы. Если мы выполняем много бизнес-логики, мы тем самым блокируем поток, и страница кажется зависшей. Сегодня мы можем выполнять код в отдельном потоке с помощью Web Workers. Однако это не поддерживается в старых браузерах, что еще больше снижает производительность на старых и медленных устройствах.

Кроме того, неконтролируемая среда выполнения также имеет определенные последствия. На сервере мы можем выбрать версию нашего языка программирования (например, JDK 7). Однако существует невероятное количество веб-браузеров в самых разных версиях. Даже самая новая версия браузера не поддерживает все указанные функции. Поэтому сложность тестирования приложения намного выше.

Таким образом, перенос маршрутизации, шаблонов и логики представления на клиента увеличивает его ответственность. Кроме того, требуется дополнительная инфраструктура, координация, дублирование и косвенность, и мы переносим код из контролируемой среды выполнения в неконтролируемую среду выполнения.

Альтернативная архитектура

Давайте не будем делать шаг назад и сравним архитектуру из предыдущего раздела с классическим приложением с графическим интерфейсом, которое мы будем разрабатывать с помощью .NET, Swing или Eclipse RCP. Мы поймем, что различия довольно малы. Некоторым командам это кажется очень привлекательным, особенно если они еще не разработали никаких веб-приложений. Опыт работы с трехуровневой архитектурой можно перенести в сеть. Средой выполнения больше не является Windows с CLR или JVM, а среда выполнения JavaScript в браузере.

К сожалению, несмотря на свою популярность, этот подход игнорирует реальную силу сети, которая была сформирована декларативными языками и поэтому имеет чрезвычайно устойчивую к ошибкам архитектуру. Это можно проиллюстрировать изменением технологии: браузер представляет собой высокопроизводительный графический движок и является наиболее оптимизированным приложением практически на всех платформах. Он реализован на C, C++ или других низкоуровневых языках, таких как Rust, и максимально использует аппаратное ускорение базовых систем. Мы можем использовать эти возможности, обратившись к ним через API из JavaScript. В качестве альтернативы мы можем использовать декларативные языки HTML и CSS, которые были разработаны специально для этой среды. Низкоуровневый код браузера может парсить,

Поскольку эти языки были созданы именно для этой цели, их производительность оптимальна. Кроме того, они декларативны и особенно

устойчивы к ошибкам в неблагоприятной среде выполнения браузера, описанной выше. Если в коде JavaScript есть ошибка, генерируется исключение, и остальная часть кода не будет выполняться. CSS, с другой стороны, пропускает неизвестные правила, а HTML рассматривает неизвестные теги как элементы. Их декларативный характер также позволяет в значительной степени обеспечить совместимость между версиями браузеров. Вот как страницы HTML и CSS, отображаемые на стороне сервера, могут интерпретироваться старыми и новыми браузерами, и даже устройства, которые не существовали во время создания кода, обычно могут интерпретировать их в той степени, в которой они пригодны для использования.

Те, кто пишет код на стороне сервера, могут рассматривать HTML как конфигурацию компонентов: можно использовать существующие компоненты, такие как текстовые поля и поля ввода, элементы выбора (списки), а также видео и автоматические элементы, без их повторной реализации. Естественно, JavaScript также используется, но он больше не является необходимым компонентом рабочей страницы, а вместо этого является дополнительным элементом, который может улучшить эргономику или внешний вид страницы.

Мы рекомендуем этот подход, основанный на основных функциях Интернета и использовании этих трех ключевых технологий на основе этих основных принципов. По этой причине мы создали веб-сайт с лучшими практиками и дали подходу имя: ROCA (ROCA означает ресурсно-ориентированную клиентскую архитектуру).

Как я могу применить это на практике?

До сих пор мы говорили об абстрактной архитектуре, но как это выглядит на практике? Можем ли мы представить себе веб-приложение без JavaScript? Возможно, мы привыкли получать некоторую структуру JSON с сервера и использовать ее для создания элементов DOM. Но JSON — это не язык Интернета. HTML — это язык Интернета. Почему нам нужно сделать дополнительный шаг, чтобы среда JavaScript могла создавать HTML из нашего JSON? Почти каждый серверный фреймворк включает механизм шаблонов, с помощью которого мы можем напрямую генерировать HTML и отправлять его клиенту. Например, приложение Spring-Boot по умолчанию поставляется в комплекте с Thymeleaf, и мы можем использовать его для прямой доставки HTML-страниц без использования какой-либо инфраструктуры JavaScript.

Заставьте это работать: HTML

Если мы хотим разработать приложение без JavaScript, первым шагом будет создание веб-приложения, состоящего только из HTML. HTML5 включает довольно много элементов с большим количеством функций. Вот три примера:

<input type="date">создает средство выбора даты.

<audio>создает плеер для аудиофайлов

<video>создает проигрыватель для видеофайлов.

Кроме того, для этих элементов всегда есть запасной вариант, чтобы старые браузеры могли продолжать работать. Часто браузер может принимать более правильные решения о том, как отображать эти элементы, чем мы. Приложение <input type="date">на смартфоне оптимизировано таким образом, что пользователи могут выбирать дату пальцем, и по этой причине использовать этот инструмент выбора даты проще, чем инструмент выбора даты из библиотеки JavaScript. Мы также не должны забывать, что HTML предназначен не только для логики представления. С помощью простой ссылки мы можем перейти с одной страницы на другую. HTML-формы также чрезвычайно эффективны и позволяют нам создавать динамические HTTP-запросы на основе пользовательского ввода, которые затем отправляются на сервер. Затем сервер выполняет изменение, запрошенное пользователем (рис. 4).

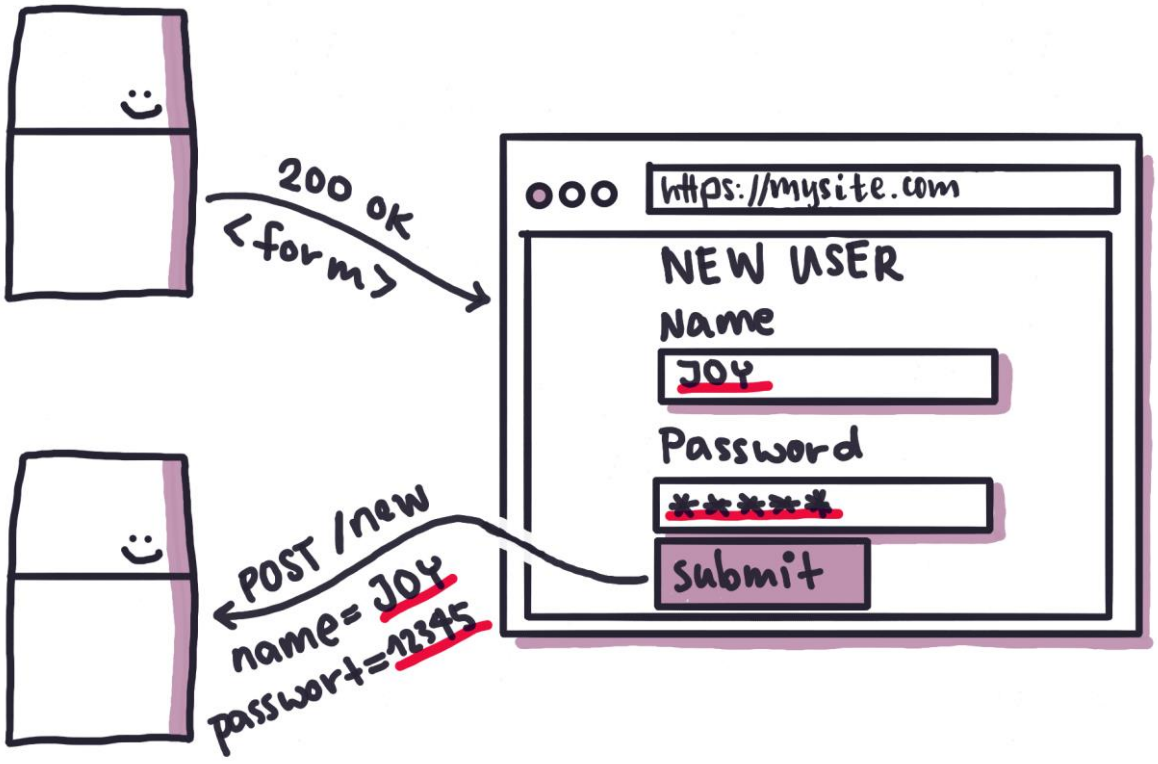

Рисунок 4. HTML-формы позволяют выполнять динамические HTTP-запросы

Иногда мы также хотим показать другую страницу в зависимости от ввода пользователя. Поскольку сервер теперь контролирует функциональность в приложении, он может решить, куда идти дальше с помощью перенаправления (рис. 5).

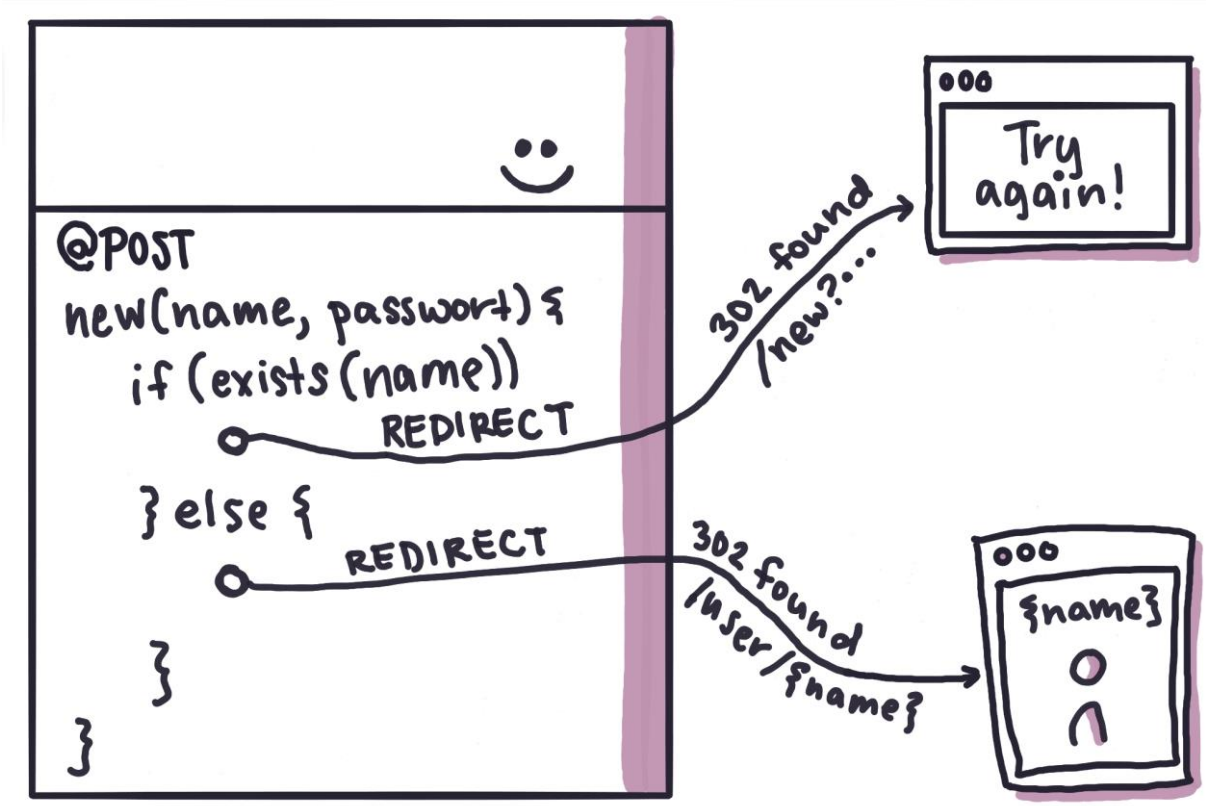

Рисунок 5. Сервер решает, что делать дальше

Одним из основных преимуществ этого подхода является то, что HTML может легко использоваться программами чтения с экрана. Это позволяет относительно легко реализовать специальные возможности в приложении. Проект A11Y — хороший ресурс для дальнейших улучшений. Конечно, также возможно создать доступное приложение, ориентированное на JavaScript, но это требует больших усилий и большого внимания.

Возможность повторного использования

Еще одним преимуществом HTML является то, что мы можем произвольно вкладывать элементы. Чтобы упростить повторное использование элементов пользовательского интерфейса, мы можем извлечь некоторые элементы в компоненты. Компоненты — это многократно используемая структура HTML, которую мы можем использовать в качестве абстракции для нашей функциональности.

Чтобы поддерживать обзор наших компонентов, мы можем структурировать их в системе, подобной Atomic Design. Подробнее о них можно узнать в статье Уте Майер на стр. 58. Здесь более крупные компоненты всегда содержат более мелкие компоненты. Самый простой способ применить это на практике — определить фрагмент HTML, скопировать разметку и отключить содержимое в компоненте. Тем не менее, это не легко поддерживать с течением

времени. Большинство движков шаблонов предоставляют способ определить фрагмент HTML один раз и сделать его многоразовым. В Thymeleaf мы можем сделать это с помощью th:replaceи th:insert.

Одним из основных преимуществ этого подхода является также то, что мы можем стилизовать наши компоненты на основе этой HTML-структуры. Здесь начинаются более серьезные проблемы.

Сделать красиво: CSS

Теперь мы подошли к моменту, когда мы можем потерпеть неудачу. С одной стороны, можно утверждать, что важнее всего, чтобы приложение работало, даже если оно выглядит не очень красиво. С другой стороны, никто не хочет использовать уродливое приложение. В этом контексте важна определенная эстетика.

Эстетика для Интернета

Хотя «красота» субъективна, мы можем потратить много времени на такие темы, как дизайн или типографика, чтобы развить чувство того, как все должно выглядеть в готовом продукте. К сожалению, у многих разработчиков нет необходимого времени или желания по-настоящему вникать в тему.

К счастью, нам не всегда приходится изобретать велосипед. Можно использовать готовые библиотеки компонентов вроде Bootstrap или Foundation . Здесь дизайнеры уже приняли самые важные решения, чтобы приложение было удобно использовать конечным пользователям.

Для тех, кто хочет изменить библиотеку компонентов и расширить ее дополнительными компонентами, очень полезен набор инструментов библиотеки шаблонов, такой как Fractal . Это особенно полезно, когда компоненты должны повторно использоваться между проектами. Для больших проектов CSS мы рекомендуем использовать препроцессор, такой как Sass, чтобы CSS можно было разделить на несколько файлов. Затем CSS для определенного компонента HTML можно разместить непосредственно рядом с компонентом в той же папке. Здесь мы можем использовать соглашение об именах, такое как БЭМ для компонентов CSS. Конвейер активов, такой как faucet-pipeline , может использоваться для компиляции CSS. Когда мы также пишем JavaScript для нашего приложения, мы также можем использовать кран для этого.

Сделайте это быстро: JavaScript

Мы достигли точки, на которой можно остановиться: у нас есть работающее веб-приложение, которое выглядит красиво. Однако теперь мы можем оптимизировать и улучшить наше приложение с помощью JavaScript. Причина, по которой SPA часто кажется быстрее, заключается в том, что браузеру не нужно

анализировать и оценивать CSS и JavaScript при каждом переходе страницы. С такой библиотекой, как Turbolinks , мы можем добиться именно этого: когда пользователь нажимает на ссылку, HTML на странице заменяется без перезагрузки CSS и JavaScript.

С HTML-формами мы можем добиться чего-то подобного: вместо того, чтобы отправлять браузеру содержимое формы и извлекать результат в виде новой страницы, мы можем отправить форму через Ajax и работать с возвращаемым HTML-кодом, заменяя только части страницы. .

Нет необходимости создавать одностраничное приложение, чтобы избежать повторного анализа CSS и JavaScript при каждой загрузке страницы.

Чтобы объединить содержимое нескольких страниц в браузере, мы можем трансклюдировать ссылки — это означает, что ссылка заменяется ее содержимым. Для этого некоторый общий код JavaScript может «следовать» по ссылке — например, использовать Ajax для отправки запроса на сервер — и включать результат непосредственно на страницу.

Это особенно привлекательно, потому что во всех этих трех случаях эффект аналогичен эффекту одностраничного приложения, а переходы страниц работают плавно. Однако страница работает и тогда, когда JavaScript отсутствует, глючит, еще не загружен или по каким-то другим причинам не может быть выполнен.

Для улучшения JavaScript в нашем приложении теперь мы можем использовать пользовательские элементы для определения наших собственных элементов HTML в браузере:

```
class MyComponent extends HTMLElement {
 connectedCallback() {
 // instantiate component
 }
}
customElements.define("my-component", MyComponent)
```

Преимущество здесь в том, что логика, определенная в connectedCallbackфункции, всегда будет выполняться браузером при создании <my-component>элемента в HTML DOM. Это также работает вместе с Turbolinks или включением, потому что браузер заботится о создании экземпляра JavaScript, как только элемент появляется в DOM.

СПА против РПЦ

На практике мы убедились, что когда мы разрабатываем приложения с серверным рендерингом HTML и пользовательскими элементами, тщательно модульным CSS и небольшим количеством JavaScript, мы можем создавать архитектурно чистые, компактные, быстрые и эргономичные приложения. Мы не боимся сравнения этих приложений с приложениями, созданными с использованием популярных подходов SPA — во многих аспектах они часто превосходят их. Мы обнаружили, что они не только имеют меньшие коммуникационные издержки, более высокую отказоустойчивость и более легко доступны, но и их легче поддерживать в долгосрочной перспективе.Мы зависим от браузера и его стандартов, а не от фреймворка SPA, который может полностью измениться в следующей версии или может быть заменен следующей альтернативной фреймворком.

СПИСОК ЛИТЕРАТУРЫ

1. Zachary Kessin. Programming HTML5 Applications, Building Powerful Cross-Platform Environments in Ja- vaScript. Sebastopol CA, 2011.
2. Andrew Lunny. PhoneGap Beginner's Guide. Birmingham, 2011.
3. The Node Ahead: JavaScript leaps from browser into future. The Register. [Электронный ресурс]. – Режим доступа: http://www.theregister.co.uk/2011/03/01/the_rise_and_rise_of_node_dot_js/. – Дата доступа: 05.09.2012.

4. John Paul Mueller. Professional IronPython, Wrox Professional Guides. Chichester, 2010
5. Boydlee Pollentine. Appcelerator Titanium Smartphone App Development Cookbook. Birmingham, 2011.

ОБУЧЕНИЕ ПРОГРАММИРОВАНИЯ ЧЕРЕЗ РАЗРАБОТКУ ИГР

Минуллин А.

магистрант 2 курса Международного университета Астана, г.Нур-Султан

Аннотация. Статья посвящена вопросам обучения программированию и информатике в средней школе. Информационные технологии интенсивно развиваются, что требует постоянной корректировки содержания обучения как школьников, так и студентов высших учебных заведений. Описываются проблемы образования с учетом требований рынка труда ИТ-специалистов, который предъявляет к выпускникам учебных заведений все более высокие требования. При этом подчеркивается, что проблемы существуют не только в отечественных школах, но и во всем мире. Предлагаются пути решения ряда проблем, при этом дается беглый обзор подходов, реализуемых в школах различных стран мира.

Ключевые слова: программирование, обучение, ИТ, Unity, C#

ВВЕДЕНИЕ

Традиционные методы обучения становятся менее эффективными, чем раньше. Учащиеся сегодня находят нынешнее образование скучным и трудным. А компьютерное программирование по-прежнему трудно освоить. Однако задача педагога состоит в том, чтобы удовлетворить потребности учащихся и найти наилучший способ передачи знаний, навыков и умений, необходимых учащимся для понимания программирования.

Компьютерное программирование — один из немногих предметов, где учащиеся могут использовать компьютерные игры не только как систему обучения, но и создавать свои собственные игры. Кроме того, игры и разработка игр, как правило, развивают творческое мышление, что является одной из самых сложных проблем в обучении программированию. Сегодня новое поколение учеников играет в цифровые игры больше, чем когда-либо, и обучение на основе игр кажется мотивирующим решением для преодоления основных проблем, с которыми учащиеся сталкиваются в классе — обучение программированию.

В этой статье будет описан подход к обучению программированию с помощью игр и обучению программированию с помощью разработки игр.

Изучение программирования с помощью игр

Реформа системы образования позволяет учителям использовать все методы для повышения эффективности преподавания предмета учащимся. В то же время

развитие информационных технологий дает возможность каждому учащемуся получать доступ к информации через Интернет и позволяет создавать новые формы отображения. И в то же время игры продолжают оставаться неотъемлемой частью процесса обучения.

Игры, используемые в образовательном процессе, весьма разнообразны и могут быть классифицированы по множеству различных критериев. Поскольку мы будем использовать компьютерные игры для обучения программированию, мы применим к играм следующую классификацию:

- игры для изучения основ алгоритмов;
- игры, обучающие определенному языку программирования;

Для изучения основ программирования разработаны такие ресурсы, как Lightbot и Minecraft, Code.org и др. Каждый из этих ресурсов отличается тем, что он сделан для детей от 8. Обучение начинается с основ алгоритмов, можно использовать простой язык программирования. Например, в Lightbot, игре-головоломке, нам нужно запрограммировать робота так, чтобы он выделял все синие ячейки в 3D-сетке, и нам даже не нужно писать код, как при изучении нового языка программирования, а сложность игры заключается в создании программы для робота, способного пройти путь за один подход. Если рассматривать Minecraft и Code.org, то в этих играх используется графический язык, который состоит из блоков, используя нужные блоки, пользователь определяет логику работы программы и одновременно наблюдает за работой интерпретатора. Игра построена на выполнении простых задач, но есть возможность выбора уровня задачи, которую хочет решить пользователь. Представленные игры, по сути, являются обучающими играми: они развивают пространственное, математическое и логическое мышление.

Для изучения того или иного языка программирования можно использовать известные игровые ресурсы, такие как JavaRush, Codecombat, Ruby Warrior и т. д., где предлагается изучить тот или иной язык программирования или выбрать из списка языков, поддерживаемых Игровой платформой. Например, на платформе JavaRush, как следует из названия, предлагается изучить язык программирования Java. Исследования проходят в форме онлайн-игры, где каждый уровень представляет собой отдельную солнечную систему на игровой звездной карте. Успешно завершив миссию, космический корабль перелетает из одной звездной системы в другую. Немного иначе устроена онлайн игра Ruby Warrior. Код здесь тоже написан в браузере, но на Ruby, а герой игры — 8-битный рыцарь, который должен пройти через коварные подземелья и замки, чтобы найти принцессу. И, наконец, онлайн-игра от разработчиков CodeCombat предлагает вам обучиться программированию в форме RPG (RolePlaying Game — ролевая игра). Мы можем начать обучение без какого-либо опыта программирования, плюс возможность

выбора языков программирования. Этот ресурс предлагает изучить язык программирования Python или JavaScript, а уровни усложняющейся игры делают ее довольно увлекательной и легкой. Пользователь берет на себя роль волшебника и с помощью программного кода меняет игровой мир вокруг персонажа. Для каждого уровня ставится определенная задача — например, сбор драгоценных камней, которые помогут призвать юнитов (союзников) и найти выход из лабиринта.

Изучение программирования через разработку игр

Данный метод обучения представляет собой основанный на проектах. Проектный метод обучения программированию ведется с двойной целью - для обучения программированию и попытки привлечь новые группы учащихся к разработке программ и изучению ООП.

Этот подход к обучению включает в себя задания и финальный мини-проект, построенный на идее обучения кодированию посредством разработки игр. В финальном проекте студенты должны создать свою собственную цифровую игру и объяснить основной игровой процесс. Студенты должны разрабатывать игры только в соответствии с более конкретными инструкциями, предоставленными учителем.

Основным средством обучения программированию посредством разработки игр является игровой движок. В настоящее время любой может создать видеоигру без каких-либо знаний в области программирования. Главное правильно выбрать софт. Однако это не означает, что процесс разработки станет проще. Даже такие простые игры, как Loose Birds и Tetris, требуют много времени и усилий. Но благодаря специальному программному обеспечению то, на что раньше уходили годы, теперь занимает месяцы или дни. Ниже приведен список лучших программ для разработки игр.

Если вы никогда в жизни не написали ни строчки кода, Construct 3 — это то, что вам нужно. Эта программа полностью управляется через графический интерфейс. Это означает, что все элементы можно легко создавать и перетаскивать мышью по принципу перетаскивания (drag-and-drop). Игровая логика и переменные реализованы с использованием конструктивных особенностей, предоставляемых самим приложением. Программа поставляется с сотнями ссылок, объясняющих концепцию создания игр от базового до продвинутого уровня на доступном языке.

GameMaker Studio 2 — это переписанное и многократно улучшенное приложение, созданное в 1999 году. На сегодняшний день это один из самых популярных «движков» для разработки игр, регулярно обновляемый. Как и в Construct 3, в GameMaker Studio 2 вы можете создавать полные готовые игры,

используя интерфейс перетаскивания для переменных и игровой логики. Однако, в отличие от Construct 3, GameMaker Studio 2 также может создавать сценарии с использованием языка Game Maker, довольно гибкого языка сценариев, напоминающего C.

Unity — универсальный редактор игр и кроссплатформенная среда разработки игр. В нем уществует множество инструментов, которые помогут быстро редактировать и выполнять итерации во время рабочего процесса создания игры, включая игровые режимы. Этот режим позволяет просматривать дизайн в режиме реального времени. Программа поддерживает возможности разработки как 2D-, так и 3D-игр для удовлетворения конкретных потребностей разных жанров. С помощью этой навигационной системы вы можете создавать NPC, которые помогут разумно ориентироваться в игровом мире. Предопределенные игровые объекты, такие как префабы Unity, помогают создавать эффективные и адаптируемые рабочие процессы, облегчающие разработку и программирование.

Таким образом, видно, что игры могут применяться для изучения темы «Алгоритмизация и программирование» в начальных классах школьной программы по информатике. И изучение программирования через разработку игры можно применить в старших классах для улучшения тем объектно-ориентированного программирования. Надо учесть что, введение игр в практику учителя, можно рекомендовать лишь для закрепления или контроля усвоения отдельных тем учениками, поскольку для полноценного изучения на уроке требуется достаточно много времени, что реализовать в формате школьного урока не всегда возможно.

Из этого следует, что тема «алгоритмизация и программирование» может рассматриваться в игровой форме на базовом уровне школьной программы информатики. Кроме того, исследования в области программирования посредством разработки игр могут быть применены в старшей школе для улучшения предмета объектно-ориентированного программирования. Надо помнить, что введение игры в практику учителя рекомендуется только для интеграции или контроля усвоения учеником той или иной темы, так как полноценное обучение за одно занятие требует времени и его не всегда можно реализовать в виде школьных уроков.

ЗАКЛЮЧЕНИЕ

Статья посвящена вопросам обучения программированию и информатике в средней школе. Информационные технологии интенсивно развиваются, что требует постоянной корректировки содержания обучения как школьников, так и студентов высших учебных заведений. Описываются проблемы образования с учетом требований рынка труда ИТ-специалистов, который предъявляет к выпускникам учебных заведений все более высокие требования. При этом подчеркивается, что проблемы существуют не только в

отечественных школах, но и во всем мире. Предлагаются пути решения ряда проблем, при этом дается беглый обзор подходов, реализуемых в школах различных стран мира.

СПИСОК ЛИТЕРАТУРЫ

1. Шагбазян Д.В., Штанюк А.А. Проблемы школьного образования и требования рынка труда в сфере ИТ // Наука сегодня: вызовы и решения: материалы международной научно-практической конференции. Научный центр «Диспут». 2017. С. 130-131.
2. Гладских Д.С., Штанюк А.А. О проблемах формирования компетенций в области программирования у бакалавров IT-направления // Информатика и образование. 2015. № 5. С. 71-74.
3. Винник В.К., Штанюк А.А. Информационно-проектный метод при подготовке будущих специалистов в сфере информационных технологий (с использованием системы MOODLE) // Фундаментальные исследования. 2015. № 2-23. С. 5183-5186.
4. Каган Э.М. Возможности и перспективы применения технологий и средств визуального программирования при обучении школьников // Вестник российского университета дружбы народов. Серия: информатизация образования. 2018. Т. 15. № 1. С. 18-28.
5. Горностаева Т.Н., Горностаев О.М. Обучение программированию будущих учителей информатики // Проблемы и приоритеты развития науки в xxi веке: сборник научных статей по материалам Международной научно-практической конференции. 2017. С. 86-90.
6. Юн С. IT Школа Samsung: учим школьников разработке мобильных приложений. URL: https://habr.com/ru/company/samsung/blog/413669/.
7. Сотрудничество с ИТ-компаниями. URL: http://www.itmm.unn.ru/sotrudnichestvo/sotrudnichestvo-s-it-kompaniyami/.
8. Education and training monitor 2020. Country analysis. URL: https://op.europa.eu/s/oADc (дата обращения: 25.12.2020).
9. Passey D. Computer science (CS) in the compulsory education curriculum: Implications for future research. Educ Inf Technol. 2017. V. 22. P. 421–443. DOI: 10.1007/s10639-016-9475-z.
10. Computing in the national curriculum. A guide for primary teachers. URL: http://www.computingatschool.org.uk/data/uploads/CASPrimaryComputing.pdf

ПЛЮСЫ ИЗУЧЕНИЯ ПРОГРАММИРОВАНИЯ ЧЕРЕЗ РАЗРАБОТКУ ИГР

Минуллин А.

магистрант 2 курса Международного университета Астана, г.Нур-Султан

Аннотация. Статья посвящена вопросам обучения программированию и информатике в средней школе. Информационные технологии интенсивно развиваются, что требует постоянной корректировки содержания обучения как школьников, так и студентов высших учебных заведений. Описываются проблемы образования с учетом требований рынка труда ИТ-специалистов, который предъявляет к выпускникам учебных заведений все более высокие требования. При этом подчеркивается, что проблемы существуют не только в отечественных школах, но и во всем мире. Предлагаются пути решения ряда проблем, при этом дается беглый обзор подходов, реализуемых в школах различных стран мира.

Ключевые слова: программирование, обучение, ИТ, Unity, C#

ВВЕДЕНИЕ

Ориентиры Казахстана на цифровизацию экономики требуют изменений в образовании. Современные школьные учебные программы содержат дисциплины, связанные с основами информатики, что является отражением все возрастающей потребности рынка труда в ИТ-специалистах [1–4]. Информационные технологии интенсивно развиваются, что требует постоянной корректировки содержания обучения как школьников, так и студентов высших учебных заведений. Рынок специалистов предъявляет высокие требования к уровню знаний студентов и выпускников вузов. Без владения современными информационными технологиями разработки программного обеспечения у молодых специалистов наблюдаются проблемы с трудоустройством в компании по разработке ПО [5]. Потребности рынка вакансий в высококвалифицированных ИТ-кадрах сильно опережают то, что могут предложить школы, техникумы и даже вузы [6, 7].

Ситуация с изучением программирования в школах не выглядит оптимистично [6-9]. Это справедливо не только для нашей страны, но и для развитых зарубежных стран. В школе наибольший приоритет, как и ранее, отдается как фундаментальным предметам (математике, физике), так и гуманитарным (истории, литературе). Здесь проявляются как многолетние традиции, особенно в специализированных школах, гимназиях, так и низкий уровень образования самих педагогов. Однако введение в школьные учебные программы курсов алгоритмизации и программирования является важным условием проявления интереса к данной отрасли у школьников, что значительно повлияет на выбор

будущей профессии, связанной с ИТ-сферой, и непосредственную готовность изучения данных предметов в вузе.

Целью данного исследования является анализ ситуации с ИТ-образованием в средних учебных заведениях и выработка рекомендаций по улучшению этой ситуации. Предлагаются пути решения ряда проблем, при этом дается беглый обзор подходов, реализуемых в школах различных стран мира.

Современные дети буквально рождаются с гаджетами в руках. Многие изучают все основные навыки использования смартфонов и компьютеров в детском саду. Изменились и уроки информатики в школе. Раньше основное внимание уделялось развитию пользовательских навыков, а теперь основное внимание уделяется обучению студентов программированию и новым навыкам грамотности.

Однако не все учителя проходят программирование на уроках, поэтому дети чаще узнают что-то неактуальное и не до конца примененное. Более того, в среднем школы посвящают компьютерным наукам всего два часа в неделю, от чего ученики не узнают многого.

На первый взгляд изучение языка программирования и написание программ для детей может быть сложным и скучным. Да, программа не всегда работает с первого раза. Да, это требует внимания и терпения. Но кто сказал, что сложные вещи не могут быть интересными?

Уроки программирования в школе могут быть интересными, если использовать новые технологии и создавать правильные задачи. С помощью программирования можно не только решать математические уравнения, но и разрабатывать игры. И поэтому я применил в 8-ых классах курс по разработке игры с использованием приложения Unity, которое содержит широкий набор программных разделов.

Самая сложная часть в изучении программирования — это не заучивать синтаксис кода, а понять смысл кода. Желательно, нужно найти баланс между изучением синтаксиса и его практической реализацией.

Применяя курс программирования к разработке игр, я следовал следующим шагам:

- Определение конечной цели

При изучении языка программирования в первую очередь нужно иметь хотя бы приблизительное представление о том, чему мы в конечном итоге хотим научить наших учеников. Например, наша цель — научиться разрабатывать игру. В данном курсе ученики 8-ых классов должны были создать 2D-игру платформер.

- Применение знаний на практике

Например, мы изучили часть материала о работе с объектами. Это даже может быть просто одна команда. Сразу запускаем эту команду для проверки. Программа Unity идеально подходит для этого. Unity показывает, как работает только что написанный код.

- Создание проектов

Проект — это место, где можно применить навыки, полученные с помощью заданий и теорий. Чтобы сделать программирование более интересным для студентов, я использовал проектный метод обучения. Этот метод требует от каждой команды разработки и демонстрации игры в конце курса. Знания быстро приобретаются, когда работа над программным проектом выполняется в реальной рабочей среде. Здесь большую роль играет эффект новизны, как из-за относительно недавно изученного языка программирования, так и из-за новой рабочей среды.

МАТЕРИАЛ И МЕТОДЫ ИССЛЕДОВАНИЯ

В последнее время крупные отечественные и зарубежные ИТ-компании стали взаимодействовать с учебными заведениями (например, Яндекс сотрудничает с Высшей Школой Экономики, МФТИ, СПБГУ, Mail.ru Group – с МГТУ им. Баумана, Интел и Харман – ННГУ им. Н.И. Лобачевского) [10, 11]. Как правило, на старших курсах организуются дополнительные занятия в виде факультативов, успешно зарекомендовала себя практика открытия интернатуры. Для школьников организуются курсы по робототехнике, на которых в сочетании с игровыми технологиями происходит знакомство с миром программирования и алгоритмизации. Можно считать такие курсы успешными примерами приобщения школьников к будущей профессии, формирования определенной информационной культуры. Однако очевидно, что, если не вносить изменения в программы школьных предметов, то разрыв между уровнем образования и требованиями ИТ рынка труда будет увеличиваться. Об эффективности сотрудничества с ИТ-компаниями говорят и зарубежные авторы [12].

Эффективность обучения, по нашему мнению, зависит от методик и подходов преподавания компьютерных наук в школах. Так, грамотно построенный курс программирования поможет школьникам понять основные концепции, такие как алгоритмы, языки программирования, архитектура вычислительных систем и т.д. Программирование дает школьникам возможность проявить себя в роли создателя или проектировщика; обладая знаниями языка программирования, можно написать практически любую программу. А интересные и актуальные для разработки темы помогут стимулировать развитие у детей творческих способностей. При этом навыки, получаемые школьниками в результате освоения опыта проектировщика, способствуют развитию критического мышления,

освоению приемов решения проблем в других областях. Однако школы России и многих развитых стран Запада нуждаются в реформировании учебных планов на государственном уровне [12].

Одним из подходов в этом направлении считается возможно раннее обучение школьников алгоритмизации и программированию [13–17]. В Великобритании, Франции, Ирландии разработаны курсы по программированию для учащихся начальных классов, а также проводятся дополнительные занятия для школьников, где они могут познакомиться с основами алгоритмов и созданием простейших компьютерных программ. В школах Австрии, Финляндии школьникам преподаются основы программирования, где применяют визуальную событийно-ориентированную среду Scratch (https://scratch.mit.edu).

В Южной Корее учащиеся в средней школе изучают программирование с 2015 г., оно представлено несколькими предметами цикла «Компьютерные науки»: «Информационное общество и компьютер», «Программирование», «Программное обеспечение», «Компьютерное моделирование», «Компьютерный дизайн». Причем этот цикл предметов начинается в младшей школе (с 6 лет) [18]. Учащимся старшей школы предлагаются курсы по выбору. Следует отметить, что современная Computer Science происходит из США, но только в 10% школ США имеются курсы по изучению основ программирования, а изучение компьютерных наук начинается с университетов. Для изменения этой ситуации в США были запущены агитационные и мотивационные видеоролики с участием известных программистов, основателей IT-компаний, спортсменов и звезд с целью вызвать высокую заинтересованность к изучению программирования у школьников и широкой аудитории страны. К сожалению, автор статьи не обнаружил в анализе данные по Казвхствну.

В качестве успешного примера по внедрению новых обучающих стандартов можно указать Великобританию. Осенью 2015 г. во всех школах Великобритании были внедрены учебные планы по изучению основ программирования – «Национальная учебная программа Англии: компьютерные программы обучения» [19]. Так, ученикам от 5 лет по курсу программирования предлагается изучать создание элементарных программ, а с 11 лет учащиеся будут осваивать различные алгоритмы и, по крайней мере, два языка программирования. Ученикам предлагается изучать простую логику (например, операторы AND, OR и NOT), работу с двоичными системами и совместную работу компьютерного оборудования и программного обеспечения, соответствующим образом использовать структуры данных [например, списки, таблицы, массивы]; разрабатывать модульные программы, которые используют процедуры или функции. Часто используемые языки программирования – это Java или Python. В школах Великобритании и раньше изучали основы компьютерной грамотности, но эти программы были старые и рассчитаны на изучение работы с компьютером в качестве пользователя, что никак не связано с программированием и

компьютерными науками. Все эти новые подходы и программы преподавания компьютерных наук вызывали немалый негатив со стороны организаций, чьи ИКТ стандарты были долгие годы внедрены в школах Великобритании. Новая учебная программа предполагает преподавать детскую компьютерную науку, информационные технологии и цифровую грамотность, она направлена на обучение написанию кода и созданию собственных программ. Предполагаются изучение работы с компьютером, а также изучение принципов работы компьютера и возможностей заставить его работать на себя. Введение нового обучающего стандарта в школьные курсы было связано с частыми жалобами технических компаний на нехватку квалифицированных кадров. В разработку курса были вовлечены Microsoft, Google и Королевская инженерная академия.

В Китае, учитывая растущую потребность в IT-специалистах, которые могут создавать сложные приложения и управлять ими, в образовательной системе происходит обучение программированию в самом раннем возрасте [20]. Многие китайские дети сейчас получают доступ к программированию уже перед школой. Обычно они работают над освоением математики и китайского языка, но сейчас стараются получить новые навыки и в области технологий, компьютерных наук и программирования.

Интересен опыт обучения программированию своих маленьких детей разработчиками программного обеспечения [21], основой которого являются специальная среда разработки и игровая форма обучения. Существуют специальные обучающие программы для дошкольников: Tynker, ПиктоМир, Box Island и др. Российские педагоги также делятся опытом применения такого типа программ [22, 23].

При изучении вопроса обучения программированию и изучения компьютерных наук в странах бывшего СССР интересно рассмотреть опыт Республики Армения. В Армении работают порядка 400 IT-компаний, а ежегодно запускаются десятки новых. Также в стране есть уже несколько школ, в которых дети изучают востребованные на рынке программирование, веб-дизайн, кино, видеопроизводство и компьютерные игры. Страна успешно движется в сторону массового внедрения IT-дисциплин во все школьные программы. Здесь следует отметить центр «Айб-Синхрон» и Центр креативных исследований «ТУМО» [24]. Школу «Айб-Синхрон» часто называют школой XXI века. От обычных учебных заведений она отличается высокой технической оснащенностью и специально разработанной программой. Подобные центры дополнительного образования работают во многих городах Армении. Организаторы подчеркивают, что это не просто кружки по интересам: в них рождаются идеи будущих стартапов. В армянских школах по инициативе Союза компаний по информационным технологиям Армении (UITE) реализуется программа «Робототехника начинается со школы» [25], в рамках которой задействовано 55 кружков робототехники для учеников 5–12-х классов. Как отмечает министерство образования и науки

Армении, действующие в армянских школах кружки робототехники могут стать новым образовательным брендом. Программа «Робототехника начинается со школы» осуществляется UITE при сотрудничестве с сотовым оператором VivaCell-MTS, благотворительным фондом «Кронимет», организациями World Vision Armenia и Counterpart International Armenia. По мнению руководителя общественной организации «Союз предприятий информационных технологий», если во всех школах страны будут внедрены «инженерные кабинеты», то к 2025 г. в Армении будет не менее 70 000 хороших специалистов, работающих в самых разных сферах ИТ и конкурентоспособных на мировом рынке. В стране уже в каждой пятой школе страны открыли инженерные лаборатории, где дети с ранних лет могут изучать основы программирования и робототехники.

РЕЗУЛЬТАТЫ ИССЛЕДОВАНИЯ

В рамках примененного курса группа учеников была дважды протестирована, чтобы показать, что предложенный подход позволяет с большой эффективностью обучать студентов аспектам и особенностям программирования в среде Unity. Всего в экспериментальную группу вошли 45 обучающихся.

В ходе тестирования было проверено две основные области: основы программирования Unity C# и основы Unity. На вступительных испытаниях выявляется исходный уровень знаний учащихся перед прохождением курса прикладного программирования Unity. После того, как студенты изучили курс прикладного программирования Unity, был проведен итоговый тест. Результаты двух тестов представлены в Таблице 1.

Таблица 1 – Средний балл учащихся 8-х классов до и после изучения темы курса программирования с использованием приложения Unity

	Всего учащихся	8D		8E		8F	
		девочки	мальчики	девочки	мальчики	девочки	мальчики
	45	8	7	6	9	3	12
Основы Unity 3D	Входное тестирование (15 баллов)	4,8	5,3	3,9	3,5	4,6	5,6
	Итоговое тестирование (15 баллов)	6,2	7,5	7,9	7,4	7,5	8
Основы Unity C#	Входное тестирование (15 баллов)	5,5	6,2	4,3	4,5	6,3	7,5
	Итоговое тестирование (15 баллов)	9	8,6	7,2	7,3	9,4	9,5

Для формирующего этапа теста был рассчитан средний балл по каждому показателю, представленный в таблице 2.

Таблица 2 – Средние баллы, определенные в ходе тестирования учащихся 8-х классов до и после обучения курса

№	Исследуемые параметры, характеризующие уровень знаний учащихся	Входное тестирование	Итоговое тестирование
1.	Основы Unity 3D	4,6	7,4
2.	Основы Unity C#	5,7	8,5
3.	Обобщённый средний балл	5,15	7,95

Учитывая обобщенные средние баллы по результатам проверки уровня знаний до и после курса, можно сделать вывод, что методика обучения программированию с использованием разработанного в ходе обучения приложения Unity эффективна. Хотя сложность итогового экзамена была выше сложности входного, разница между средним баллом этого показателя между входным и итоговым экзаменом составила 2,8 балла.

На самом деле, изучение программирования и создание игр для учащихся дает много преимуществ. Основные из них:

Повышение мотивации к обучению. Программирование и разработка игр — это интересный новый мир, который открывает перед детьми совершенно иной аспект знаний. Например, дети перестают спрашивать, зачем им математика или физика. О роли английского в жизни программиста говорить не стоит. Она очевидна. Все современные языки разработки требуют знания английского языка. То есть ваш ребенок намного быстрее выучит иностранный язык, что также необходимо в современном мире.

Развивать логическое мышление. Программирование и логика — неразделимые понятия. Мы не сможем написать приложение, не используя логическое мышление. Например, чтобы создать простую игру, придется продумать последовательность действий и выстроить определенные логические последовательности. В результате ребенок учится структурировать свои мысли, анализировать события, систематизировать задачи, строить планы, лучше понимать отношения между разными предметами и действиями.

Возможность самореализации. К сожалению, сегодня многие дети используют устройства впустую и готовы часами просматривать ленты социальных сетей. Благодаря разработке игры мы можем показать, насколько захватывающим может быть мир программирования. Ребенок иметь возможность развиваться, написав несколько приложений, пусть и в игровой форме.

Развитие творческих навыков. Программирование — это больше, чем просто написание кода. Разработчики творчески подходят к решению определенных задач в процессе создания приложений и игр. Один и тот же фрагмент кода можно написать по-разному, оптимизируя приложение и делая его развертывание более простым и функциональным. То есть ребенок учиться мыслить нелинейно и применять все свои знания в любой сфере человеческой деятельности.

Программирование помогает в профориентации. Программирование пригодится в жизни всем. Не все захотят его сразу использовать, а тем более становиться IT-специалистами. Полезно будет попробовать курс программирования для детей хотя бы для того, чтобы понять, насколько подходит детям эта сфера деятельности. Это позволит избежать синдрома «третьего курса» – это когда студент понимает, что ошибся с выбором профессии, уже отучившись половину срока.

Технологии 21 века развиваются с космической скоростью. Поэтому у нас появляется все больше и больше возможностей их использовать. В отличие от технологий, школьная программа не меняется каждый год. Кроме того, хотя частные уроки программирования для детей распространены, они не дают достаточных знаний и рабочей нагрузки. Учителя информатики не должны программировать как профессионалы. Это не их работа. Но освоить новую технологию и язык программирования, чтобы доносить его потом для детей, необходимо.

ЗАКЛЮЧЕНИЕ

Статья была посвящена вопросам обучения программированию и информатике в средней школе. Информационные технологии интенсивно развиваются, что требует постоянной корректировки содержания обучения как школьников, так и студентов высших учебных заведений. Описываются проблемы образования с учетом требований рынка труда ИТ-специалистов, который предъявляет к выпускникам учебных заведений все более высокие требования. При этом подчеркивается, что проблемы существуют не только в отечественных школах, но и во всем мире. Предлагаются пути решения ряда проблем, при этом дается беглый обзор подходов, реализуемых в школах различных стран мира.

СПИСОК ЛИТЕРАТУРЫ

11. Рудычева Н. Рынок ИТ-специалистов: правила диктуют соискатели // GlobalCIO. URL: https://globalcio.ru/discussion/682/.
12. Колесникова К. России может не хватить 2 миллионов IT-специалистов // Российская газета. 2018. № 26 (7489). URL: https://rg.ru/2018/01/31/kadrovyj-golod-rossii-cherez-10-let-ostanetsia-bez-it-specialistov.html
13. В России возрастает дефицит IT-специалистов, и бизнес берёт инициативу в свои руки. URL: https://propostuplenie.ru/article/v-rossii-vozrastaet-deficit-it-specialistov-i-biznes-beryot-iniciativu-v-svoi-ruki/
14. Шагбазян Д.В., Штанюк А.А. Проблемы школьного образования и требования рынка труда в сфере ИТ // Наука сегодня: вызовы и решения: материалы международной научно-практической конференции. Научный центр «Диспут». 2017. С. 130-131.
15. Гладских Д.С., Штанюк А.А. О проблемах формирования компетенций в области программирования у бакалавров IT-направления // Информатика и образование. 2015. № 5. С. 71-74.
16. Винник В.К., Штанюк А.А. Информационно-проектный метод при подготовке будущих специалистов в сфере информационных технологий (с использованием системы MOODLE) // Фундаментальные исследования. 2015. № 2-23. С. 5183-5186.
17. Каган Э.М. Возможности и перспективы применения технологий и средств визуального программирования при обучении школьников // Вестник российского университета дружбы народов. Серия: информатизация образования. 2018. Т. 15. № 1. С. 18-28.
18. Горностаева Т.Н., Горностаев О.М. Обучение программированию будущих учителей информатики // Проблемы и приоритеты развития науки в xxi веке: сборник научных статей по материалам Международной научно-практической конференции. 2017. С. 86-90.
19. Юн С. IT Школа Samsung: учим школьников разработке мобильных приложений. URL: https://habr.com/ru/company/samsung/blog/413669/.
20. Сотрудничество с ИТ-компаниями. URL: http://www.itmm.unn.ru/sotrudnichestvo/sotrudnichestvo-s-it-kompaniyami/.
21. Education and training monitor 2020. Country analysis. URL: https://op.europa.eu/s/oADc (дата обращения: 25.12.2020).
22. Passey D. Computer science (CS) in the compulsory education curriculum: Implications for future research. Educ Inf Technol. 2017. V. 22. P. 421–443. DOI: 10.1007/s10639-016-9475-z.
23. Computing in the national curriculum. A guide for primary teachers. URL: http://www.computingatschool.org.uk/data/uploads/CASPrimaryComputing.pdf
24. Computing in the national curriculum. A guide for secondary teachers. URL: http://www.computingatschool.org.uk/data/uploads/cas_secondary.pdf

25. Босова *Л.Л.* Школьная информатика в России и в мире // Информатизация образования и науки. 2018. № 3 (39). С. 134-145.
26. Босова *Л.Л.* Современные тенденции развития школьной информатики в России и за рубежом // Информатика и образование. 2019. № 1 (300). С. 22-32.
27. Маняхина В.Г. Южнокорейский подход к обучению информатике в младшей и средней школе // Проблемы современного образования. 2015. № 6. С. 59-67. URL: http://www.pmedu.ru/images/pso6/Pages%20from%20PSO2015-6-6.pdf
28. National curriculum in England: computing programmes of study. URL: https://www.gov.uk/government/publications/national-curriculum-in-england-computing-programmes-of-study/national-curriculum-in-england-computing-programmes-of-study
29. В Китае программирование становится «новым английским». URL: https://hightech.fm/2017/05/30/china-10.