Муниципальное бюджетное общеобразовательное учреждение

«Гимназия № 2» города Сарова

***Изучение принципов построения графических приложений с помощью библиотеки JavaFX на примере реализации разных математических задач.***

**Выполнил**: ученик 10 класса «Б»

МБОУ Гимназии №2

Аристов Сергей Антонович

**Руководитель**: Гудкова Евгения Александровна

учитель информатики

МБОУ Гимназии №2

**Оглавление**

[​ Введение 3](#__RefHeading___Toc263_2223766980)

[​ Теоретическая часть 4](#__RefHeading___Toc241_1050172711)

[​ Отбор технологий 5](#__RefHeading___Toc265_2223766980)

[AWT 5](#__RefHeading___Toc267_2223766980)

[Swing 5](#__RefHeading___Toc269_2223766980)

[JavaFX 7](#__RefHeading___Toc271_2223766980)

[​ Архитектурное изучение 7](#__RefHeading___Toc273_2223766980)

[SOLID-принципы 7](#__RefHeading___Toc275_2223766980)

[MVC-паттерны 7](#__RefHeading___Toc277_2223766980)

[Принцип «Чистого кода» 7](#__RefHeading___Toc279_2223766980)

[​ Выбор задачи 7](#__RefHeading___Toc281_2223766980)

[Клеточные автоматы и игра «Жинь» 7](#__RefHeading___Toc283_2223766980)

[Фазовое пространство 7](#__RefHeading___Toc285_2223766980)

[​ Глава 2. Практическая часть 7](#__RefHeading___Toc243_1050172711)

# Введение

Программирование одно из ведущих направлений трудовой деятельности в современном мире. Навыки математически-программного моделирования высоко оцениваются при найме на работу, особенно когда такие модели выглядят наглядно и понятно.

Решение задач программного проектирования и математического моделирования послужило идеей моего проекта по нескольким причинам:

* Навыки архитектурного проектирования ПО необходимы и высокооплачиваемы при найме в компанию на вакансию «программист»
* Понимание принципов модульного взаимодействия компонентов благоприятствует эффективности решения различных задач вообще, не только специфичных задач программирования
* Личный интерес, меня всегда интересовало, как можно красиво и не слишком сложно создавать красивую графику, на понятном мне языке, таком как Java, особенно мне интересно программировать необычные абстрактные математические концепции, такое как фазовое пространство

Задачи, которые я ставлю перед собой представлены:

* Изучить литературу и видеоматериалы по темам:
  + Общие принципы построения графического пользовательского интерфейса(далее – ГПИ)
  + Графические инструменты, доступные для языка Java
  + Материалы по некоторым математическим объектам, таких как клеточные автоматы, или фазовое пространство
* Провести отбор технологий
* Придумать несколько вариантов архитектуры, с последующим выбором наилучшего варианта
* Написать итоговое приложение
* Подвести итоги проделанной работы

# Теоретическая часть

## Отбор технологий

В этой части мы разберемся какие существуют технологии для вывода приложения на экран, и постараемся выбрать лучшую

### AWT

Начнём изучение этой темы в хронологическом порядке. Для языка Java, который был выпущен в 1995 году, с самого начала[[1]](#endnote-2) существовала стандартная реализация некоторых графических компонентов, на основе которых можно было написать графическое приложение. Она называлась Java AWT(Abstract Window Toolkit — англ. Абстрактный Оконный Инструментарий).

Она(библиотека) использовала связку java-кода с нативным кодом, написанным на C++.

Эта библиотека является очень тяжеловесной, и часто не удобной в использовании, к тому же, в силу того, что она не сама рисует компоненты на экране, а лишь использует системные вызовы, программы, написанные с использованием AWT работают медленно, и нестабильно.

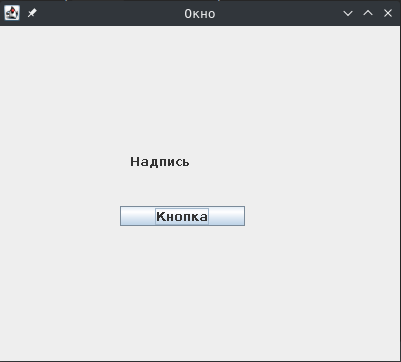
На данный момент она считается устаревшей, и начинать новые проекты на ней не стоит.

### Swing

На смену Java AWT почти сразу пришел Java Swing.[[2]](#endnote-3)(Версия языка 1.2). У Swing очень похожая архитектура на AWT, но с достаточно серьезными отличиями, компоненты Java Swing «Легковесны», т. е для отрисовки себя используют исключительно безопасный java-код, не задействуя на прямую системные вызовы, из-за этого графические программы стали быстрее и стабильнее, а так же такая архитектура позволила писать более гибкие программы, потому что весь код был написан на java, так что любой разработчик мог свободно подстраивать даже уже написанные компоненты под себя, к тому же, теперь программы на java стали выглядеть унифицированно, похоже друг на друга, не сильно завися от данной оконной системы.

Приведен пример кода с использованием Java Swing, и результат его работы

*package ru.cmr;  
import javax.swing.\*;  
public class SwingExample {  
 public static void main(String[] args) {  
 JFrame window = new JFrame("Окно");  
 JLabel label = new JLabel("Надпись");  
 JButton btn = new JButton("Кнопка");  
  
 label.setBounds(130, 110, 100, 50);  
 btn.setBounds(120, 180, 125, 20);  
  
 window.setLayout(null);  
 window.setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);  
 window.setBounds(300,300, 400,360);  
 window.add(label);  
 window.add(btn);  
 window.setVisible(true);  
 }  
}*

 *Результат работы кода, приведенного выше.*

Так выглядит типичное приложение Java Swing, видно что код достаточно понятный, и результат можно получить достаточно быстро, но внешний вид достаточно устаревший.

### JavaFX

Хоть Swing был во многом лучше AWT, но все равно он быстро устаревал, и было решено разработать альтернативную технологию, которая будет и удобной, и красивой, и производительной. Эту технологию назвали JavaFX. Согласно сайту компании Oracle[[3]](#endnote-4) JavaFX имеет ряд преимуществ:

* **Java APIs** – JavaFX – это Java-библиотека, так что её собственный код хорошо взаимодействует с другими Java-библиотеками, а так же с JVM-языками, вроде Groovy, Kotlin, JRuby, Jython, и тд..
* **FXML, SceneBuilder** – Внешний вид JavaFX-приложения можно описывать в специальном формате, называемым FXML, а не прописывать все детали прямо в коде программы.
* **WebView** – в стандартную поставку JavaFX входит компонент, использующий WebKitHTML технологии для отрисовки внутри себя web-страницы. Что делает возможным с легкостью сделать, допустим, свою реализацию браузера, или сделать всё приложение — как комплекс web-страниц, а часть приложения на Java, запускает эти страницы.
* **Swing-совместимость** – JavaFX может спокойно работать со Swing, использовать уже написанные на Swing компоненты, а существующие приложения на Swing могут быть постепенно перестроены на JavaFX.
* **3D Графика** – существенное различие между Swing и JavaFX — это возможность использовать 3D графику.
* **Аппаратно-ускоренный процесс отрисовки** – JavaFX использует технологию, которая использует графический ускоритель, для отрисовки компонентов на экране.
* **Самодостаточная модель распространения приложения –** приложения на JavaFX распространяются в виде так называемых «слепков», другими словами, приложение распространяется в виде запускаемого файла для конкретной системы, и при запуске, запускается виртуальная машина, которая запускает само приложение, и эта виртуальная машина не связана с работой других виртуальных машин в системе(если, к примеру, работает сервер, он не будет затронут нашим приложением).

#### Архитектура

Многие возможности этой библиотеки существуют благодаря её архитектуре.

Это изображение взято с официального сайта компании Oracle[[4]](#endnote-5). Как видно за т. н. «публичном интерфейсе» скрыта целая сложная система, позволяющая создавать красивые и производительные приложения.



## Архитектурное изучение

### SOLID-принципы

### MVC-паттерны

### Принцип «Чистого кода»

## Выбор задачи

### Клеточные автоматы и игра «Жинь»

### Фазовое пространство

# Глава 2. Практическая часть

Для создания практической части проекта я использовала программу MS PowerPoint.

**2.1. Создание электронной викторины в PowerPoint**

Для создания викторины в PowerPoint необходимо:

1. Запустить программу PowerPoint. Для этого выполните Пуск/ Все программы/ Microsoft Office/ PowerPoint (рис. 4).

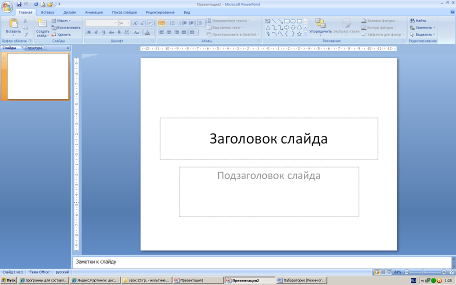
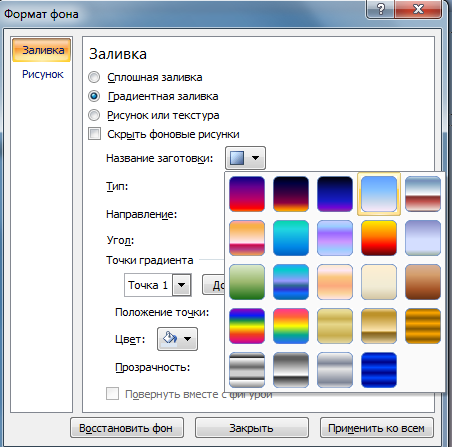
 

Рис.4 Рис.5

1. Выбрать цветовое оформление слайдов (рис. 5). Вкладка Дизайн/ Стили фона/ Формат фона/Градиентная заливка/ Рассвет/ Применять ко всем (рис. 6).

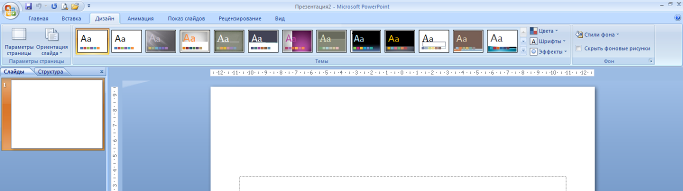


Рис.6

1. Оформить заголовок. Вкладка Вставка/ WordArt (рис. 7).

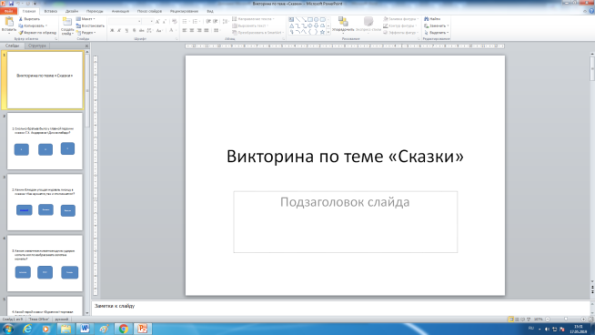


Рис.7

1. Зайти во Вкладку Дизайн/ Стили фона/ Формат фона/Рисунок или текстура нажать на слово Файл выбрать путь Рабочий стол /Презентации/Практическая работа№2/картинку фон (рис. 8)

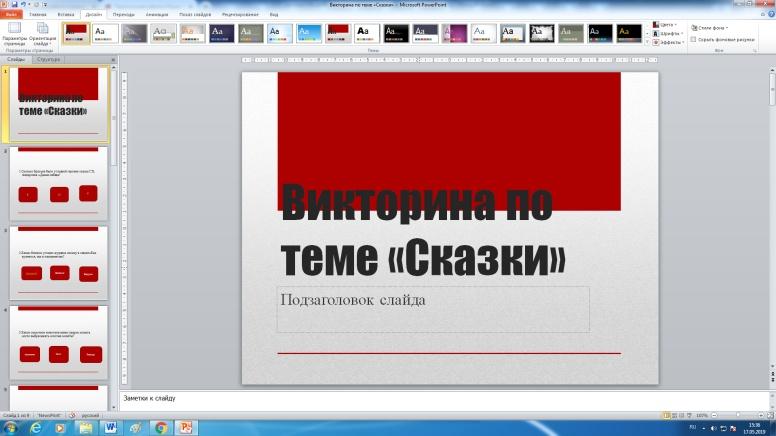


Рис.8

1. 2 слайд Главная/ Создать слайд/Два объекта. Набрать текст (рис. 9)

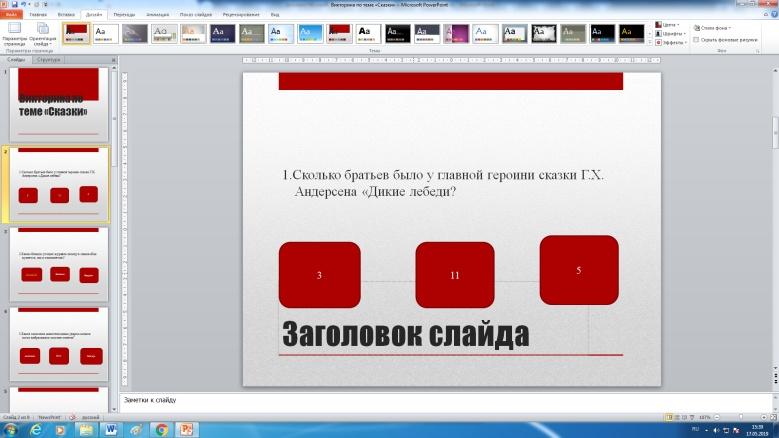


Рис.9

1. Остальные слайды делать аналогично 2 слайду
2. Возвратиться ко второму слайду для создания гиперссылок (рис. 10)

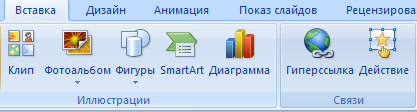


Рис.10

1. Выделить слово. Выбрать вкладку Вставка-Гиперссылка. Появляется диалоговое окно выбрать Место в документе и слайд и нажать ОК. Аналогично для остальных слайдов/ (рис. 11)

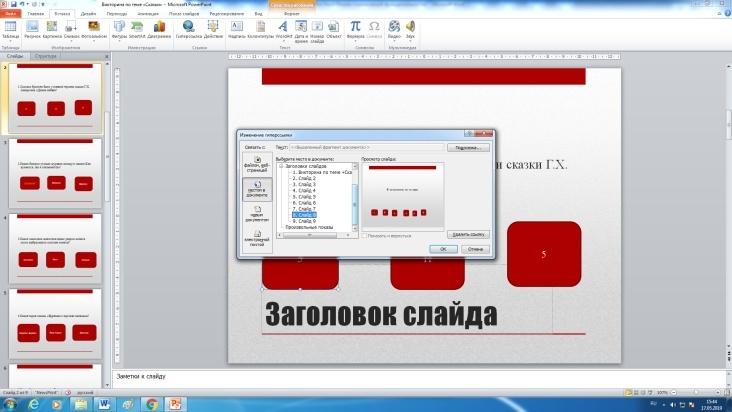


Рис.11

1. Перейти на второй слайд выбираем Вставка - Фигуры (рис. 12)

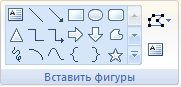


Рис.12

1. Настроить гиперссылки. Выделить кнопку. Выбрать в меню Вставка-Гиперссылка (рис. 13)

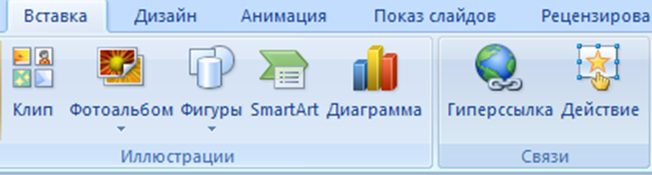


Рис.13

1. Аналогично для остальных слайдов вставка управляющих кнопок и создание гиперссылок
2. Создать последний пустой слайд с помощью объекта WordArt вставляем надпись «Спасибо за внимание!»

**Заключение**

В ходе работы над индивидуальным проектом была достигнута поставленная цель, которая заключалась в создании электронных викторин в среде MS PowerPoint.

Для реализации поставленной цели мною были достигнуты задачи:

1) При выполнении работы были рассмотрены основные виды компьютерных викторин, представленные в среде Интернет;

2) Проанализировано программное обеспечение для создания компьютерных викторин, этапы их разработки.

3) Описана технология разработки компьютерных викторин.

4) Созданы компьютерные викторины для учащихся начальной школы.

Применение компьютерных викторин позволяет дифференцировать процесс обучения школьников с учетом их индивидуальных особенностей, дает возможность творчески работающему учителю расширить спектр способов предъявления учебной информации, позволяет осуществлять гибкое управление учебным процессом, является социально значимым и актуальным.

**Список литературы**

1. Грамолин В.В. Обучающие компьютерные игры. // Информатика и образование. – 2014. № 4. – С. 63–65.

2. Зубрилин А.А. Игровой компонент в обучении информатике. // Информатика в начальном образовании. – 2016. № 3. – С. 3–78.

3. Камалов Р.Р. Компьютерные игры в школе. // Информатика и образование. – 2015. № 2. – С. 36–48.

4. Макасер И.Л. Игра как элемент обучения. // Информатика в начальном образовании. –2018. № 2. – С. 71– 77.

5. Овчинникова С.А. Игра на уроках информатики. // Информатика и образование. – 2019. № 11. – С. 89–93.

6. Первин С.П. Дети, компьютеры и коммуникации. // Информатика и образование. – 2014. № 4. – С. 17–20.

7. Цветаева М.С. Методические материалы по игровым технологиям. // Методический вестник. – 2018. № 8. – С. 12–33.

8. Презентация-викторина "Открывайка-ка - угадай-ка" по иллюстрациям из знакомых сказок. Часть 2. (3-5 класс) [Электронный ресурс]. Режим доступа: load/196-1-0-20655, дата обращения: 01.05.2019

9. Как сделать презентацию на компьютере в Microsoft PowerPoint [Электронный ресурс]. Режим доступа: microsoft-office/office-2010/503-kak-sdelat-prezentaciyu-na-kompyutere-v-microsoft-powerpoint.html, дата обращения:16.04.2019

10. Потенциал программы Power Point как средства создания интерактивных дидактических материалов для уроков информатики [Электронный ресурс]. Режим доступа: files/conf2011/2/Kulikova\_Bobrovskaya.pdf, дата обращения: 03.05.2019

1. Oracle: официальный сайт компании-владельца языка Java [Электронный ресурс] – URL (Дата обращения 12.04.21)

   https://docs.oracle.com/javase/8/docs/api/index.html [↑](#endnote-ref-2)
2. Oracle: страница документации [Электронный ресурс] – URL (Дата обращения 12.04.21)

   https://docs.oracle.com/javase/8/docs/api/index.html?overview-summary.html [↑](#endnote-ref-3)
3. Oracle: официальная статья по JavaFX [Электронный ресурс] — URL (Дата обращения 18.04.21)

   https://docs.oracle.com/javase/8/javafx/get-started-tutorial/jfx-overview.htm [↑](#endnote-ref-4)
4. Oracle: статья по архитектуре JavaFX [Электронный ресурс] – URL (Дата обращения 18.04.21)

   https://docs.oracle.com/javase/8/javafx/get-started-tutorial/jfx-architecture.htm#CHDFDAFF [↑](#endnote-ref-5)