Министерство образования и науки РФ

Государственное образовательное учреждение высшего профессионального образования

«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ ТОМСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»

Институт (факультет) – Кибернетический Центр

Направление (специальность) – Информатика и вычислительная техника

Выпускающая кафедра – Автоматики и Компьютерных Систем

«**СПРАЙТОВАЯ ИГРА НА HTML5 «АРКАНОИД»**»

Пояснительная записка к курсовой работе

по дисциплине «Технология разработки программного обеспечения»

|  |  |  |
| --- | --- | --- |
| Студент гр.8В94 | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ (подпись)  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ (дата) | А.С. Лизин |
| Преподаватель | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ (подпись)  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ (дата) | П.И. Банокин |

Томск – 2012

# Задание

В ходе выполнения курсовой работы должно быть создано игровое приложение, представляющее собой игру Арканоид.

Выполнение курсовой работы состоит из создания программного приложения согласно варианту и создания документации, включающей: требования к программному продукту, оформленные в виде SRS (приложение А), описание архитектуры приложения с использованием следующих диаграмм: class, use-case, component, deployment, activity, sequence.

# Реферат

Курсовой проект 34 страниц, 11 рисунков, шесть источников, одно приложение.

Ключевые слова: игра, JavaScript, HTML5, Арканоид.

Цель работы – создание клиентского веб-приложения с использованием объектно-ориентированного подхода. В первой части пояснительной записки приводится выявление требований к программному продукту.

Во второй части описывается архитектура программной системы.

В третьей части производится детализация архитектуры.

В четвертой части описывается интерфейс пользователя.

В пятой части описывается процесс тестирования.

Продуктом разработки является игра типа «Арканоид».

Пояснительная записка к курсовому проекта написана в текстовом редакторе Мicrosoft Word 2010. Приложение было разработано в среде JetBrains WebStorm 4.0.2 на языке JavaScript. Диаграммы создавались в программах StarUML, MSPaint.

# Содержание

Оглавление

[Задание 1](#_Toc342776026)

[Реферат 2](#_Toc342776027)

[Содержание 3](#_Toc342776028)

[Введение 4](#_Toc342776029)

[Основная часть 5](#_Toc342776030)

[1. Выявление требований программному продукту 5](#_Toc342776031)

[1.1 Документ SRS 5](#_Toc342776032)

[1.2 Диаграмма вариантов использования 5](#_Toc342776033)

[1.2.1 Вариант использования «Управлять ракеткой» 6](#_Toc342776034)

[1.2.2 Вариант использования «Посмотреть скорость» 6](#_Toc342776035)

[1.2.3 Вариант использования «Посмотреть количество попыток» 6](#_Toc342776036)

[2. Общее описание архитектуры программной системы 7](#_Toc342776037)

[2.1 Тип приложения и применяемый архитектурный стиль 7](#_Toc342776038)

[2.2 Шаблоны проектирования 7](#_Toc342776039)

[2.3 Диаграмма развертывания 8](#_Toc342776040)

[2.4 Диаграмма компонентов 9](#_Toc342776041)

[3. Детализация архитектуры 10](#_Toc342776042)

[3.1 Диаграмма классов 10](#_Toc342776043)

[3.2 Диаграмма активности 12](#_Toc342776044)

[3.3 Диаграммы последовательностей 13](#_Toc342776045)

[4. Описание интерфейса пользователя 16](#_Toc342776046)

[5. Тестирование 18](#_Toc342776047)

[Заключение 21](#_Toc342776048)

[Список использованных источников 22](#_Toc342776049)

[Приложения 23](#_Toc342776050)

# Введение

Существует несколько подходов к разработке программного обеспечения: процедурно-ориентированный, структурно-ориентированный, объектно-ориентированный, функционально-ориентированный и логико-ориентированный.

Объектно-ориентированный подход основан на объектной декомпозиции задачи, абстракции объекта в виде класса, типизации и наследовании объектов.

При выполнении данного курсового проекта будет разработано клиентское веб-приложение выполняющееся в браузере с использованием объектно-ориентированного подхода. Процесс разработки сопровождается составлением документации в виде документа SRS, описания программы, а также различных диаграмм, сопровождающих процесс проектирования программного обеспечения.

# Основная часть

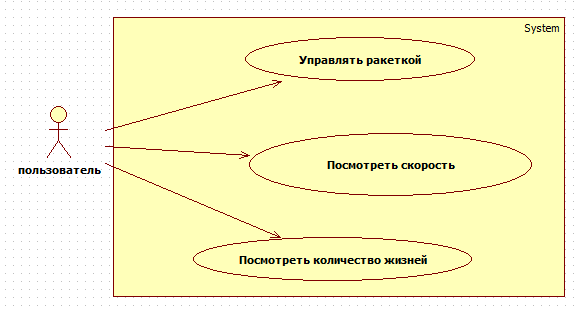
В ходе проектирования программного продукта были созданы UML – диаграммы ниже будет приведенная диаграмма вариантов использования остальные диаграммы можно посмотреть в приложении.

# Выявление требований программному продукту

## Документ SRS

Предъявляемые требования к программному продукту отражены в документе SRS, документ доступен в приложении А.

## Диаграмма вариантов использования



*Рисунок 1 – Диаграмма вариантов использования*

Пользователь может управлять ракеткой, просматривать текущую скорость шарика и количество попыток.

### Вариант использования «Управлять ракеткой»

Вариант использования состоит в возможности игрока управлять шариком с помощью стрелок «влево» и «вправо» на клавиатуре. Вариант использования доступен сразу после старта и в ходе работы приложения, но не доступен, если игра закончилась победой или проигрышем.

### Вариант использования «Посмотреть скорость»

Пользователю доступен индикатор текущей скорости шарика, которая зависит от числа собранных фишек.

### Вариант использования «Посмотреть количество попыток»

Пользователю доступен индикатор текшего количества попыток. В начале игры дается 3 попытки и игра кончается когда число попыткок станет равно нулю.

# Общее описание архитектуры программной системы

## Тип приложения и применяемый архитектурный стиль

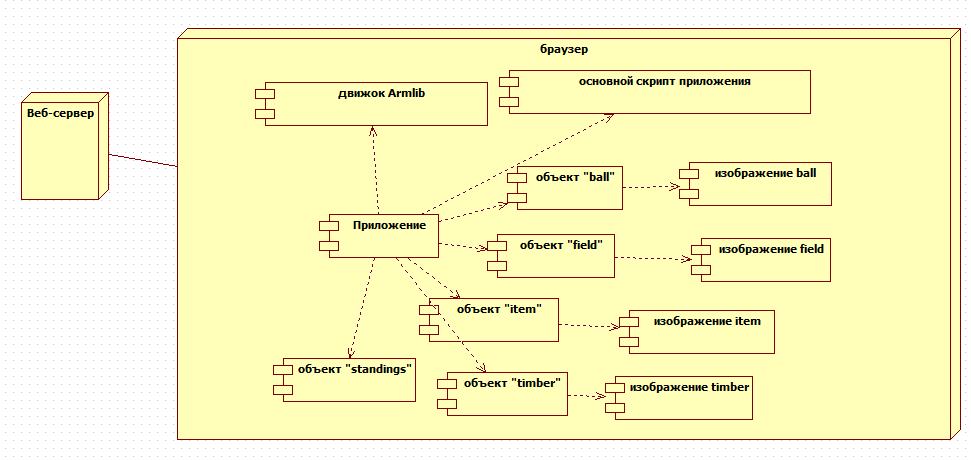
Приложение является клиентским веб-приложением выполняющимся в браузере пользователя. Приложение имеет клиент-серверную архитектуру.

## Шаблоны проектирования

В приложении использованы порождающие шаблоны проектирования: «Декоратор», « Абстрактная фабрика» и один поведенческий шаблон «Наблюдатель». Подробнее о применении шаблонов можно посмотреть в пункте 3.1 Диаграмма классов.

## Диаграмма развертывания

Диаграмма развертывания отображает аппаратные элементы компьютера, другие устройства и программные компоненты, а также процессы и объекты, которые им назначены. Диаграмма показана на рисунке 2.

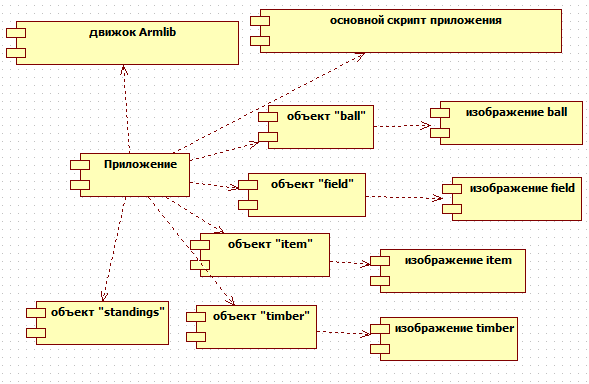


*Рисунок 2 – Диаграмма развертывания*

Из диаграммы можно видеть, что приложение выполняется в браузере клиента и загружается с сервера.

## Диаграмма компонентов

На диаграмме компонентов отображаются компоненты программного обеспечения и связи между ними. Диаграмма показана на рисунке 2.



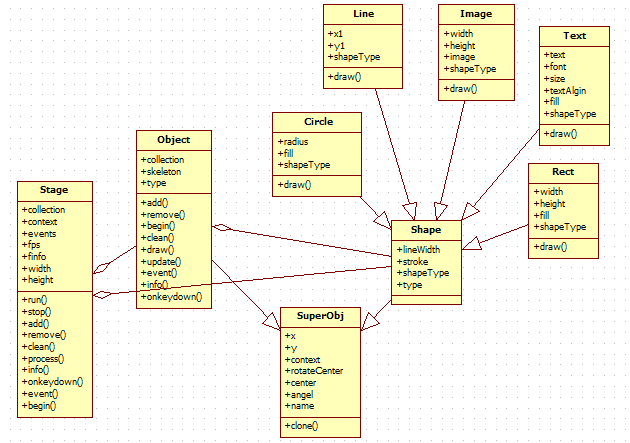
*Рисунок 3 – Диаграмма компонентов*

Приложение разделено на компоненты игры и компонент движка. Объект ball и описывает шарик, объект timber описывает ракетку, объект item описывает фишки, которые нужно сбивать шариком, объект standings описывает индикаторы попыток и текущей скорости шарика. Выше перечисленные объекты являются объектами класса CObject.

# Детализация архитектуры

## Диаграмма классов

Перед началом разработки приложения была создана диаграмма классов (рисунок 4).



*Рисунок 4 – Диаграмма классов*

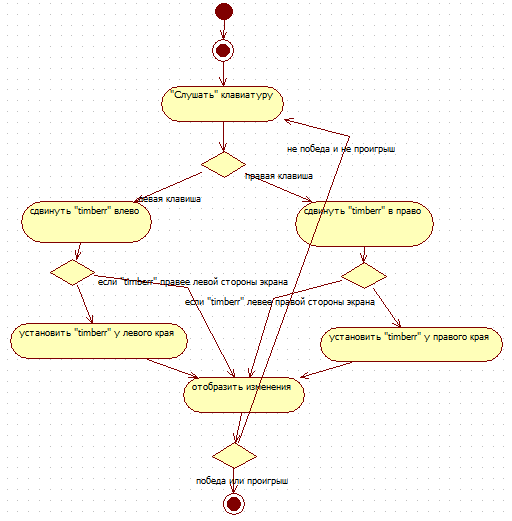
Данная диаграмма отображает отношения между классами движка. Класс Stage, является менеджером движка, т.е. через его методы происходит управление всем движком. Можно видеть, что класс Stage и Object агрегирует в себе объекты классов Object и классы производные от класса Shape. Данное отношение между классами позволяет ввести абстракцию объектов, т.е. структурные части игры представлены не просто набором классов, функций и т.д., а одним игровым объектом. Данный подход позволяет уменьшить связность и сделать структурные части игры более наглядными. Классы Rect, line, Circle, Text, Image описывают графические примитивы, с помощью которых рисуются игровые объекты.

В движке были использованы шаблоны проектирования такие как:

* Шаблон «Декоратор» использован в классах графических примитивов: Rect, line, Circle, Text, Image. В данных классах данный шаблон дает возможность более удобно работать с API контекста рисования HTML5.
* Шаблон «Абстрактная фабрика» использован в классах: Stage, Object, Rect, Cline, Circle, Text, Image. В данных классах данный шаблон дает возможность задать поведение объектов при их создании.
* Шаблон «Наблюдатель» использован в классе Stage и используется для извещения всех агрегированных объектов о изменении объекта класса Stage.

## Диаграмма активности

Так же была разработана диаграмма активности для детализации основных прецедентов в игре (рисунок 5).

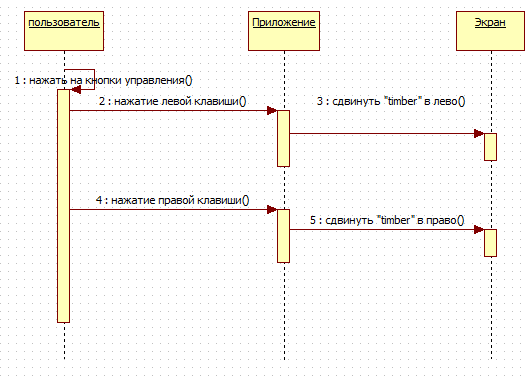
****

*Рисунок 5 – Диаграмма активности для процесса управления ракеткой*

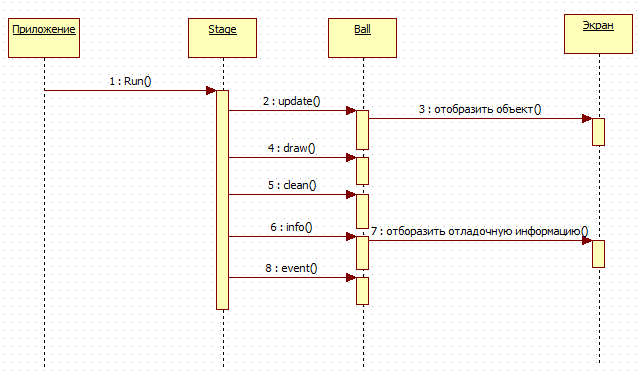
С помощью диаграмм последовательностей были показаны взаимодействия объектов, упорядоченные по времени их проявления.

## Диаграммы последовательностей

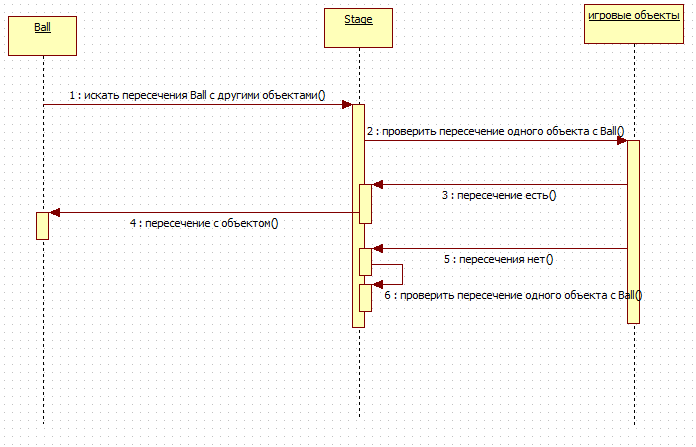
С помощью диаграммы последовательности были показаны взаимодействия объектов, упорядоченные по времени их проявления (рисунок 6).



*Рисунок 6 – Диаграмма последовательности для процесса управления ракеткой*

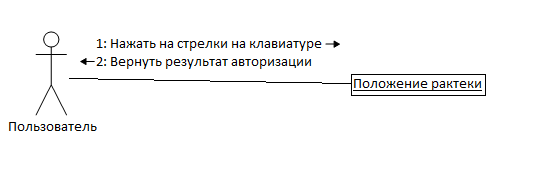
****

*Рисунок 7 – Диаграмма последовательности для обновления и отображения объекта ball*

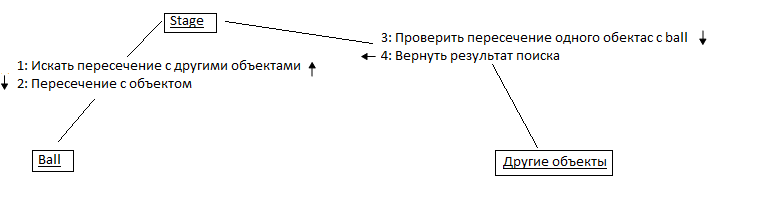


*Рисунок 8 – Диаграмма последовательности поиска пересечения объекта ball с другими игровыми объектами*

С помощью диаграммы кооперации были отображены следующие действия: перемещения ракетки по экрану, поиск пересечения шарика с другими игровыми объектами (рисунок 9, 10).



*Рисунок 9 – Диаграмма кооперации для перемещения ракетки по экрану.*



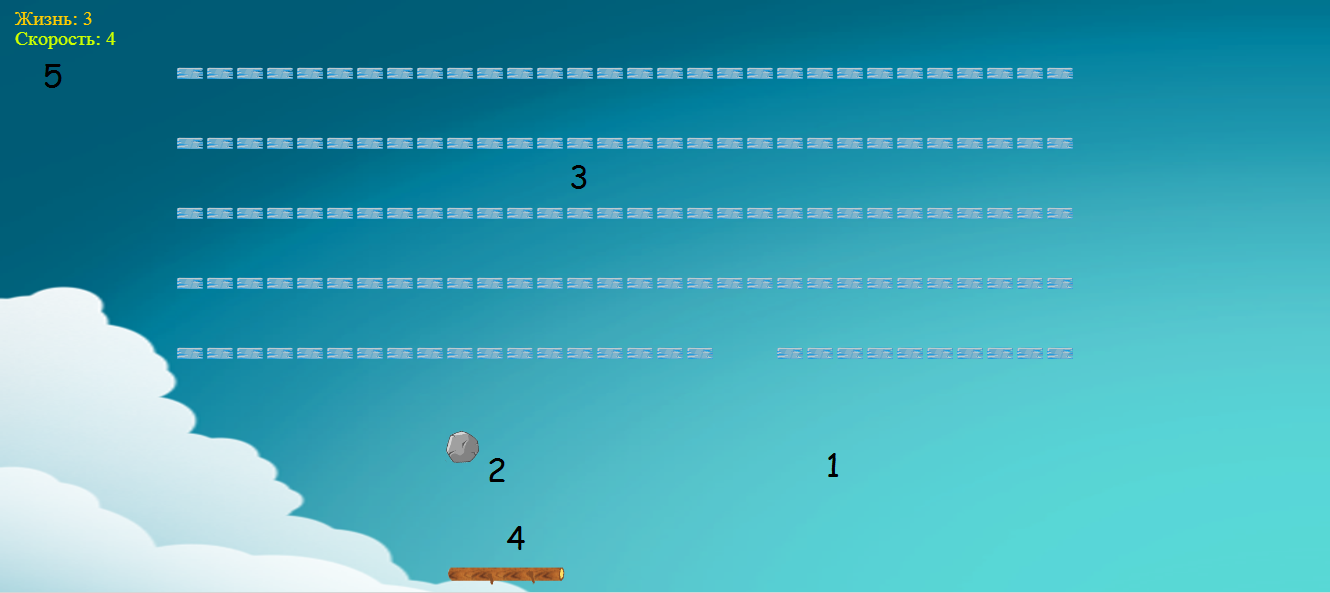
*Рисунок 10 – Диаграмма кооперации для поиска пересечения объекта ball с другими игровыми объектами.*

# Описание интерфейса пользователя

После запуска игры пользователь может видеть (рисунок 11):

1. Игровое поле изображающие небо.
2. Движущийся по полю шарик.
3. Несколько десятков фишек в виде изображений льдин.
4. Ракетку, представленную изображением бревна.
5. Индикаторы количества попыток и текущей скорости шарика.

При нажатии на стрелки «влево» или «вправо», пользователь может перемещать по экрану ракетку, с помощью чего может отбивать шарик когда, тот летит в сторону нижней части экрана. В случае победы или проигрыша, пользователь не может управлять ракеткой. Победа в игре происходит, если были сбиты все фишки и число попыток больше нуля. Если пользователь не отбил шарик и тот улетел за нижний край экрана, у пользователя отнимается попытка, но если число попыток становится равно нулю, то наступает проигрыш. В случае победы пользователь видит синюю надпись «Победа», в случае проигрыша красную надпись «Вы проиграли!».



*Рисунок 11 – Внешний вид игры через некоторое время после старта.*

# Тестирование

Для тестирования приложения были составлены следующие тест кейсы:

|  |  |  |
| --- | --- | --- |
| Проверка исчезновения фишки при встречи с шариком | | |
| Действие | Ожидаемый результат | Результат теста |
| Дождаться встречи шарика с любой фишкой | - Главное окно приложения остается открытым. - Название окна – «Арканоид».  - Фишка, с которой происходит встреча шарика исчезает с экрана. - Значение в поле скорость увеличивается на единицу. | пройден |

|  |  |  |
| --- | --- | --- |
| Проверка изменения направления движения шарика при встречи с ракеткой | | |
| Действие | Ожидаемый результат | Результат теста |
| Дождаться встречи шарика с ракеткой | - Главное окно приложения остается открытым. - Название окна – «Арканоид».  - Шарик изменяет направление своего движения по оси Y на противоположное. | пройден |

|  |  |  |
| --- | --- | --- |
| Изменения направления движения шарика при встрече с левым краем экрана | | |
| Действие | Ожидаемый результат | Результат теста |
| Дождаться встречи шарика с левым краем экрана | - Главное окно приложения остается открытым - Название окна – «Арканоид» - Направление движения шарика по оси X, изменяется на противоположное. | Пройден |

|  |  |  |
| --- | --- | --- |
| Проверка реакции на проигрыш | | |
| Действие | Ожидаемый результат | Результат теста |
| Дождаться когда количество попыток станет равным нулю | - Главное окно приложения остается открытым. - Название окна – «Арканоид». - Значение в поле «Попытки» равно нулю.  - Посередине экрана видна красная надпись крупным шрифтом «Вы проиграли!». | пройден |

|  |  |  |
| --- | --- | --- |
| Проверка реакции на выигрыш | | |
| Действие | Ожидаемый результат | Результат теста |
| Сбить все фишки, с помощью шарика | - Главное окно приложения остается открытым. - Название окна – «Арканоид». - Значение в поле «Попытки» больше нуля.  - Посередине экрана видна синяя надпись крупным шрифтом «Победа!» | пройден |

# Заключение

Результатом данной работы стало веб-приложение, являющееся спрайтовой игрой типа «Арканоид». Реализованное для работы в современных браузерах с поддержкой технологии HTML5. Приложение является простой демонстрацией работы графического движка ArmLib-0.6.0. Приложение имеет простой дизайн и удобный интерфейс. Данное приложение не имеет зависимостей в виде браузерных плагинов.

# Список использованных источников

1. HTML5ROCKS! [Электронный ресурс]. – режим доступа: http://www.html5rocks.com/ru/ – 11.08.12.

2. zencoder. [Электронный ресурс]. – режим доступа: http://zencoder.proessential-js-namespacing/ – 25.08.12.

3. nokarma.org. [Электронный ресурс]. – режим доступа: http://nokarma.org/2011/02/02/javascript-game-development-the-game-loop/index.html – 25.08.12.

4. Волшебство Git. [Электронный ресурс]. – режим доступа: http://www-cs-students.stanford.edu/~blynn/gitmagic/intl/ru/ch06.html – 25.08.12.

5. Флэнаган Д. JavaScript. Подробное руководство. – Пер. с англ. – СПб: СимволПлюс, 2008. – 992 с.

6. Стефанов С. JavaScript. Шаблоны. – Пер. с англ. – СПб: СимволПлюс, 2011. – 272 с.

# Приложения