СХІДНОУКРАЇНСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ
ІМЕНІ ВОЛОДИМИРА ДАЛЯ

Навчально-науковий інститут (факультет) інформаційних технологій та електроніки
Кафедра програмування та математики

ПОЯСНЮВАЛЬНА ЗАПИСКА
до кваліфікаційної випускної роботи

освітній ступінь бакалавр
спеціальність 121 „Інженерія програмного забезпечення”
(шифр і назва спеціальності)
спеціалізація „Інженерія програмного забезпечення”

на тему „ Розробка веб-додатку з ігровими функціями ”

Виконав: студент групи ІПЗ-16д _____ _С.С. Новодран _____
(підпис) (ініціали і прізвище)

Керівник _____ _В.О. Лифар _____
(підпис) (ініціали і прізвище)

Завідувач кафедри _____ _В.О. Лифар_____
(підпис) (ініціали і прізвище)

Рецензент_____

Сєвєродонецьк – 2020

СХІДНОУКРАЇНСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ
ІМЕНІ ВОЛОДИМИРА ДАЛЯ

Навчально-науковий інститут (факультет) інформаційних технологій та електроніки
Кафедра програмування та математики

Освітній ступінь бакалавр
спеціальність 121 „Інженерія програмного забезпечення”
(шифр і назва спеціальності)
спеціалізація „Інженерія програмного забезпечення”
(назва спеціалізації)

ЗАТВЕРДЖУЮ
Завідувач кафедри ПМ,
д.т.н., доцент
_____ Лифар В.О.
“____” _____2020 року

З А В Д А Н Н Я
НА КВАЛІФІКАЦІЙНУ ВИПУСКНУ РОБОТУ СТУДЕНТУ

Новодран Сергій Сергійович
(прізвище, ім’я, по батькові)

1.Тема роботи Розробка веб-додатку з ігровими функціями.

Керівник роботи доцент, д.т.н. Лифар Володимир Олексійович ,
(прізвище, ім’я, по батькові, науковий ступінь, вчене звання)
затверджений наказом університету від “___”_____20__року №___

2. Строк подання студентом роботи 20 травня 2020 р.

3. Вихідні дані до роботи Об'єктом даної роботи є процес створення веб-додатку з ігровими функціями.

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити) Вступ. Аналітичний огляд, з висвітленням наступних питань: Поняття об'єктної моделі стосовно JavaScript, основи веб-розробки, класифікація ігор розміщення коду на HTML-сторінці. Основна частина, в якої висвітлити: збір вимог та проектування, етап даталогічного проектування. Висновки. Перелік використаних джерел.

5. Перелік графічного матеріалу (з точним зазначенням обов’язкових креслеників)

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв

7. Дата видачі завдання__30 березеня 2020 року._____

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів виконання кваліфікаційної випускної роботи	Строк виконання етапів	Примітка
1	Одержання завдання на виконання роботи	30.03.20	
2	Укладання і погодження з керівником плану і етапів виконання роботи	06.04.20	
3	Узагальнення даних літературних джерел, укладання розділу «Аналіз предметної галузі»	13.04.20	
4	Аналіз шляхів виконання завдання. Вибір і погодження керівником оптимального шляху	20.04.20	
5	Укладання та тестування програмного продукту	27.04.20	
6	Укладання, оформлення та погодження пояснювальної записки з керівником	04.05.20	
7	Здача готової пояснювальної записки на кафедру	12.05.20	
8	Укладання доповіді і презентації	30.05.20	

Студент _____ С.С. Новодран_____
(підпис) (ініціали і прізвище)

Керівник роботи _____ _____В.О. Лифар_____
(підпис) (ініціали і прізвище)

ЛИСТ ПОГОДЖЕННЯ І ОЦІНЮВАННЯ

дипломної роботи студента гр. ІПЗ-16д Новодран С.С.

Науковий керівник

Доцент, к.т.н. _____ Лифар В.О.

Оцінка наукового керівника: _____

Рецензент _____

ПІБ, місто роботи, посада

Оцінка рецензента: _____

Кінцева оцінка за результатами захисту:

_____

Голова ЕК

_____ Лифар В.О.

підпис

РЕФЕРАТ

Робота містить: 41 сторінок основного тексту, 15 сторінок додатків, 7 рисунків, 14 використаних джерел.

Метою випускної кваліфікаційної роботи є розробка ігрового програми на HTML додатків із застосуванням мови JavaScript.

У роботі було досліджено технології побудови розробки гри у веб-додатку.

Система реалізована відповідно всім вимогам технічного завдання. Зроблено детальний опис процесу розробки системи.

ЗМІСТ

ВСТУП

Актуальність досліджень: Мова програмування, як і будь-який інший мову (природний, наприклад), призначений для комунікації, тобто зв'язку між що говорить і слухає. У програмуванні промовистою є програміст, а слухачем - інтерпретатор мови, деяка комп'ютерна програма, що розуміє цю мову і виконує дії відповідно до того, що вона зрозуміла. Був час, коли вважалося, що для полегшення спілкування з комп'ютером необхідно створити мову, досить близький до природного. Ця ідея, в кінцевому рахунку, не витримала випробувань часом, хоча і породила кілька прекрасних мов програмування. Часто буває, що побічні ефекти якоїсь діяльності перевершують очікування.

Об’єкт досліджень: Розробка гри у веб-додатку.

Мета дослідження: розробка ігрового програми на HTML додатків із застосуванням мови JavaScript.

Завдання дослідження:

1. Проаналізувати літературу по темі дослідження та суміжних тем
2. Розглянути основні теоретичні поняття JavaScript
3. Розробити додаток із застосуванням JavaScript

Методологічна та теоретична основа дослідження: методи розробки гри у веб-додатку.

Методи дослідження: технології побудови розробки гри у веб-додатку.

Практичне значення отриманих результатів: Одержаний результат можливо використовувати як доповнення до сайту з іграми.

Рішення поставлених завдань вимагало залучення таких методів дослідження: аналіз, розробка програми.

З розвитком цифрових технологій комп'ютери все більше вливаються в життя людини. Якщо раніше ЕОМ використовувалися виключно для складних

математичних обчислень, то сьогодні сфера їх застосування істотно розширилася. Комп'ютерні ігри - одне з найбільш масових застосувань електронних обчислювальних машин.

Розвиток ігрової індустрії йшло стрімким темпом, і особливо користувалося популярністю у підлітків. Перші ігри відрізнялися простотою інтерфейсу і логіки, але з часом вони ставали все складніше і складніше, над їх створенням працював вже не одна людина, а ціла команда розробників.

Сучасні ігри вимагають досить великої продуктивності від комп'ютера, і не кожна офісна машина в силах відтворювати їх. Однак для відпочинку від монотонної роботи часто досить простий, що не вимогливою до техніки, гри. Саме такий розробці присвячений даний курсовий проект - гра «Робокрокодил».

РОЗДІЛ 1. АНАЛІЗ РОБОЧОЇ ОБЛАСТІ

1.1 Основні поняття

HTML (Hyper Text Markup Language) - це мова розмітки документа, що описує форму відображення інформації на екрані комп'ютера. При створенні документа часто доводиться виділяти будь-яку частину тексту напівжирним шрифтом, змінювати розмір або колір шрифту, вирівнювати текст по центру сторінки і т. Д. У текстовому редакторі для цього досить виділити потрібний фрагмент і застосувати до нього форматування. Наприклад, щоб позначити текст курсивом, потрібно виділити його і натиснути кнопку «Курсив».

Мовою HTML той же ефект досягається наступним рядком коду: <i> Текст </ i> Символ <i> вказує, що текст треба виділити, починаючи з цього місця, а </ i> відзначає кінець виділеного фрагмента. Символи <i> і </ i> прийнято називати тегами. За допомогою тегів описується вся структура документа. Теги виділяються кутовими дужками "<" і ">", між якими вказується ім'я тега. Більшість тегів є парними, так як є відкриває тег (<i>) і відповідний йому закриваючий (</ i>). Закриває тег відрізняється наявністю косою риси ("/") перед його ім'ям.

Є так само теги, які взагалі не мають закриває тега, наприклад, тег розриву рядків
. Деякі теги можуть мати параметри (іноді їх називають атрибутами). Параметри вказуються після імені тега через пропуск в форматі параметр = "значення". Якщо параметрів кілька, то вони перераховуються через пробіл. Теги можуть вкладатися один в одного. Наприклад: <p> <i> Текст </ i> </ p> При вкладенні тегів необхідно дотримуватися послідовність їх закриття. Наприклад, такий код використовувати не можна: <p> <i> Текст </ p> </ i>

Назва "JavaScript" є власністю Netscape. Реалізація мови, здійснена розробниками Microsoft, офіційно називається Jscript. Версії JScript сумісні (якщо бути зовсім точним, то не до кінця) з відповідними версіями JavaScript, тобто JavaScript є підмножиною мови JScript.

JavaScript стандартизований ECMA (European Computer Manufacturers Association - Асоціація європейських виробників комп'ютерів). Відповідні стандарти носять назви ECMA-262 і ISO-16262. Ці стандарти вимагають визначається мова ECMAScript, який приблизно еквівалентний JavaScript 1.1. Відзначимо, що не всі реалізації JavaScript на сьогодні повністю відповідають стандарту ECMA. В рамках даного курсу ми у всіх випадках будемо використовувати назву JavaScript.

1.2 Поняття об'єктної моделі стосовно JavaScript

Для створення механізму управління сторінками на клієнтській стороні було запропоновано використовувати об'єктну модель документа. Суть моделі в тому, що кожен HTML-контейнер - це об'єкт, який характеризується трійкою:

- властивості;
- методи;
- події.

Об'єктну модель можна представити як спосіб зв'язку між сторінками і браузером. Об'єктна модель - це представлення об'єктів, методів, властивостей і подій, які присутні і відбуваються в програмному забезпеченні браузера, у вигляді, зручному для роботи з ними коду HTML і вихідного тексту сценарію на сторінці. Ми можемо з її допомогою повідомляти наші побажання браузеру і далі - відвідувачеві сторінки. Браузер виконає наші команди і відповідно змінить сторінку на екрані.

Об'єкти з однаковим набором властивостей, методів і подій об'єднуються в класи однотипних об'єктів. Класи - це описи можливих об'єктів. Самі об'єкти з'являються тільки після завантаження документа браузером або як результат роботи програми. Про це потрібно завжди пам'ятати, щоб не звернутися до об'єкта, якого немає.

Властивості

Багато HTML-контейнери мають атрибути. Наприклад, контейнер якоря <A ...> ... </A> має атрибут HREF, який перетворює його в гіпертекстове посилання:

<A HREF=intuit.htm> intuit </A>

Якщо розглядати контейнер якоря <A ...> ... </A> як об'єкт, то атрибут HREF буде задавати властивість об'єкта "якір". Програміст може змінити значення атрибута і, отже, властивість об'єкта:

document.links [0] .href = "intuit.htm";

Не у всіх атрибутів можна змінювати значення. Наприклад, висота і ширина графічної картинки визначаються по першій завантаженої в момент відображення сторінки зображенні. Усі наступні картинки, які замінюють початкову, змінюють масштаб до неї. Справедливості заради слід зауважити, що в Microsoft Internet Explorer розмір картинки може змінюватися.

Для спільності картини властивостями в JavaScript наділені об'єкти, які не мають аналогів в HTML-розмітці. Наприклад, середовище виконання, звана об'єктом Navigator, або вікно браузера, яке є взагалі найстаршим об'єктом JavaScript.

Методи

У термінології JavaScript методи об'єкта визначають функції зміни його властивостей. Наприклад, з об'єктом "документ" пов'язані методи open (),

write (), close (). Ці методи дозволяють згенерувати або змінити зміст документа. Наведемо простий приклад:

```
function hello ()

{

id = window.open ( "", "example", "width = 400, height = 150");

id.focus (); id.document.open ();

id.document.write ( "<H2> Привіт! </ H2>");

id.document.write ( "<HR> <FORM>");

id.document.write ( "<INPUT TYPE = button VALUE = 'Закрити вікно'");

id.document.write ( "onClick = 'window.opener.focus ();

window.close (); '> ");

id.document.close ();

}
```

У цьому прикладі метод open () відкриває потік записи в документ, метод write () здійснює цей запис, метод close () закриває потік записи в документ. Все відбувається так само, як і під час запису в звичайний файл. Якщо у вікна є поле статусу (зазвичай в ньому відображається рівень завантаження документа), то при незакритих потоці записи в документ в ньому буде "кидатися" прямокутник продовження запису, як це відбувається при завантаженні документа.

Події

Крім методів і властивостей об'єкти характеризуються подіями. Власне, суть програмування на JavaScript полягає в написанні обробників цих подій. Наприклад, з об'єктом типу button (контейнер INPUT типу button - "Кнопка") може відбуватися подія click, тобто користувач може натиснути на кнопку. Для цього атрибути контейнера INPUT розширені атрибутом обробки події click - onClick. Як значення цього атрибута вказується програма обробки події, яку повинен написати на JavaScript автор HTML-документа:

<INPUT TYPE = button VALUE = "Натиснути" onClick =

"Window.alert (' Будь-ласка, натисніть ще раз');">

Обробники подій вказуються в тих контейнерах, з якими ці події пов'язані. Наприклад, контейнер BODY визначає властивості всього документа, тому обробник події завершення завантаження всього документа вказується в цьому контейнері як значення атрибута onLoad.

Строго кажучи, кожен браузер, будь то Internet Explorer, Netscape Navigator або Opera, має свою об'єктну модель. Об'єктні моделі різних браузерів (і навіть різні версії одного) відрізняються один від одного, але мають принципово однакову структуру. Тому немає сенсу зупинятися на кожній з них окремо. Ми будемо розглядати загальний підхід стосовно до всіх браузерам, іноді, звичайно, загострюючи увагу на відмінностях між ними.

1.3 Розміщення коду на HTML-сторінці

Головне питання будь-якого початківця програміста: "Як оформити програму і виконати її?". Спробуємо на нього відповісти якомога простіше, але при цьому не забуваючи про всі способи застосування JavaScript-коду.

По-перше, виконує JavaScript-код браузер. У нього вбудований інтерпретатор JavaScript. Отже, виконання програми залежить від того, коли і як цей інтерпретатор отримує управління. Це, в свою чергу, залежить від функціонального застосування коду. У загальному випадку можна виділити чотири способи функціонального застосування JavaScript:

- гіпертекстове посилання (схема URL);
- обробник події (handler);
- підстановка (entity) (в Microsoft Internet Explorer реалізована у версіях від 5.X і вище);
- вставка (контейнер SCRIPT).

У підручниках з JavaScript опис застосування JavaScript зазвичай починають з контейнера SCRIPT. Але з точки зору програмування це не зовсім правильно, оскільки такий порядок не дає відповіді на ключове питання: як JavaScript-код отримує управління? Тобто яким чином викликається і виконується програма, написана на JavaScript і розміщена в HTML-документі.

Залежно від професії автора HTML-сторінки і рівня його знайомства з основами програмування можливі кілька варіантів початку освоєння JavaScript. Якщо ви програміст класичного спрямування (C, Fortran, Pascal і т.п.), то найпростіше починати з програмування всередині тіла документа, якщо ви звикли програмувати під Windows, то в цьому випадку починайте з програмування обробників подій, якщо ви маєте тільки досвід HTML-

розмітки або давно не писали програм, то тоді краще почати з програмування гіпертекстових переходів.

1.4 URL-схема JavaScript

Схема URL (Uniform Resource Locator) - це один з основних елементів Web-технології. Кожен інформаційний ресурс в Web має свій унікальний URL. URL вказують в атрибуті HREF контейнера A, в атрібте SRC контейнера IMG, в атрибуті ACTION контейнера FORM і т.п. Всі URL поділяються на схеми доступу, які залежать від протоколу доступу до ресурсу, наприклад, для доступу до FTP-архіву застосовується схема ftp, для доступу до Gopher-архіву - схема gopher, для відправки електронної пошти - схема smtp. Тип схеми визначається за першим компоненту URL: http://intuit.ru/directory/page.html У даному випадку URL починається з http - це і є визначення схеми доступу (схема http).

Основним завданням мови програмування гіпертекстової системи є програмування гіпертекстових переходів. Це означає, що при виборі тієї чи іншої гіпертекстового посилання викликається програма реалізації гіпертекстового переходу. У Web-технології стандартною програмою є програма завантаження сторінки. JavaScript дозволяє поміняти стандартну програму на програму користувача. Для того щоб відрізнити стандартний перехід по протоколу HTTP від переходу, програмованого на JavaScript, розробники мови ввели нову схему URL - JavaScript:

<A HREF="JavaScript:JavaScript_код"> ... </A>

<IMG SRC = "JavaScript: JavaScript_код">

В даному випадку текст "JavaScript_код" позначає програми-обробники на JavaScript, які викликаються при виборі гіпертекстового посилання в

першому випадку і при завантаженні картинки - в другому. Наприклад, при натисканні на гіпертекстове посилання.

А при натисканні на кнопку типу submit в формі можна заповнити текстове поле цієї ж форми:

<FORM NAME = f METHOD = post

ACTION = "JavaScript: window.document.fiVALUE =

'Натиснули кнопку Click'; void (0); ">

<TABLE BORDER = 0>

<TR>

<TD> <INPUT NAME = i> </ TD>

<TD> <INPUT TYPE = submit VALUE = Click> </ TD>

<TD> <INPUT TYPE = reset VALUE = Reset> </ TD>

</ TABLE>

</ FORM>

В URL можна розміщувати складні програми і виклики функцій. Варто тільки пам'ятати, що схема JavaScript працює не у всіх браузерах, а тільки в версіях Netscape Navigator і Internet Explorer, починаючи з четвертої.

Таким чином, при програмуванні гіпертекстового переходу інтерпретатор отримує управління після того, як користувач "клікнув" по гіпертекстовому посиланню.

1.5 Обробники подій

Такі програми, як обробники подій (handler), вказуються в атрибутах контейнерів, з якими ці події пов'язані. Наприклад, при натисканні на кнопку відбувається подія click:

```
<FORM> <INPUT TYPE = button VALUE = "Кнопка" on click =

"Window.alert ( 'intuit');"> </ FORM>
```

1.6 Підстановки

Підстановка (entity) зустрічається на Web-сторінках досить рідко. Проте це досить потужний інструмент генерації HTML-сторінки на стороні браузера. Підстановки використовуються в якості значень атрибутів HTML-контейнерів. [5] Наприклад, як значення за замовчуванням поля форми, що визначає домашню сторінку користувача, буде вказано URL поточної сторінки:

```
<SCRIPT>

function l ()

{

str = window.location.href;

return (str.length);

}

</ SCRIPT>
```

<FORM> <INPUT VALUE = "& {window.location.href};" SIZE = "& {l
()};">

</ FORM>

<SCRIPT>

<! - Це коментар ... JavaScript-код ... // ->

</ SCRIPT>

<BODY>

... Тіло документа ...

</ BODY>

</ HTML>

HTML-коментарі тут вставлені для захисту від інтерпретації даного фрагмента сторінки HTML-парсером в старих браузерах (у високого начальства ще зустрічаються). У свою чергу, кінець HTML-коментаря захищений від інтерпретації JavaScript-інтерпретатором (// на початку рядка). Крім того, в якості значення атрибута LANGUAGE у тега початку контейнера вказано значення "JavaScript". VBScript, який є альтернативою JavaScript - це скоріше екзотика, ніж загальноприйнята практика, тому даний атрибут можна опустити - значення "JavaScript" приймається за замовчуванням.

Очевидно, що розміщувати в заголовку документа генерацію тексту сторінки безглуздо - він не буде відображений браузером. Тому в заголовок поміщають декларації загальних змінних і функцій, які будуть потім використовуватися в тілі документа. При цьому браузер Netscape Navigator більш вимогливий, ніж Internet Explorer. Якщо не розмістити опис функції в

заголовку, то при її виклику в тілі документа можна отримати повідомлення про те, що дана функція не визначена. Наведемо приклад розміщення і використання функції:

```
<HTML>

<HEAD>

<SCRIPT>

function time_scroll ()

{

var d = new Date ();

window.status = d.getHours () + ":" + d.getMinutes () + ":" +

d.getSeconds ();

setTimeout ( 'time_scroll ();', 500);

}

</ SCRIPT>

</ HEAD>

<BODY onLoad = time_scroll ()>

<CENTER>

<H2> Годинники в рядку статусу </ H2>
```

В Internet Explorer 4.0 підстановки не підтримуються, тому користуватися ними слід акуратно. Перш ніж видати браузеру сторінку з підстановками, потрібно перевірити тип цього браузера. [8]

У разі підстановки інтерпретатор отримує управління в момент розбору браузером (компонент парсер) HTML-документа. Як тільки парсер зустрічає конструкцію & {..} у атрибута контейнера, він передає управління інтерпретатору JavaScript, який, в свою чергу, після виконання коду це управління повертає парсеру. Таким чином дана операція аналогічна підкачування графіки на HTML-сторінку.

1.7 Вставка (контейнер SCRIPT - примусовий виклик інтерпретатора)

Контейнер SCRIPT - це розвиток підстановок до можливості генерації тексту документа JavaScript-кодом. У цьому сенсі застосування SCRIPT аналогічно Server Side Includes, тобто генерації сторінок документів на стороні сервера. Однак тут ми забігли трохи вперед. При розборі документа HTML-парсер передає управління інтерпретатору після того, як зустріне тег початку контейнера SCRIPT. Інтерпретатор отримує на виконання весь фрагмент коду всередині контейнера SCRIPT і повертає управління HTML-парсеру для обробки тексту сторінки після тега кінця контейнера SCRIPT.

Контейнер SCRIPT виконує дві основні функції:

- розміщення коду всередині HTML-документа;
- умовна генерація HTML-розмітки на стороні браузера.

Перша функція аналогічна декларуванню змінних і функцій, які потім можна буде використовувати в якості програм переходів, обробників подій і підстановок. Друга - це підстановка результатів виконання JavaScript-коду в момент завантаження або перезавантаження документа.

Розміщення коду всередині HTML-документа

Власне, особливого розмаїття тут немає. Код можна розмістити або в заголовку документа, всередині контейнера HEAD, або всередині BODY. Останній спосіб і його особливості будуть розглянуті в розділі "Умовна генерація HTML-розмітки на стороні браузера". Тому звернемося до заголовку документа.

Код в заголовку розміщується всередині контейнера SCRIPT:

```
<HTML>
<HEAD>
<SCRIPT>
function time_scroll ()
{
d = new Date ();
window.status = d.getHours () + ":" + d.getMinutes () + ":"
+ D.getSeconds ();
setTimeout ( 'time_scroll ();', 500);
}
```

```
</ SCRIPT>

</ HEAD>

<BODY onLoad = time_scroll ()>

<CENTER>

<H2> Годинники в рядку статусу </ H2>

<FORM>

<INPUT TYPE = button VALUE = "Закрити вікно" onClick =
window.close ()>

</ FORM>

</ CENTER>

</ BODY>

</ HTML>
```

У цьому прикладі ми декларували функцію time_scroll () в заголовку документа, а потім викликали її як обробник події load в тезі початку контейнера BODY (onLoad = time_scroll ()).

Як приклад декларації змінної розглянемо зміна статусу вікна-нащадка з вікна-предка: створимо дочірнє вікно за допомогою наступної функції, декларувавши її, а потім і викликавши:

```
function sel ()

{
```

```
id = window.open ( "", "example", "width = 500, height = 200, status,
menu");

id.focus ();

id.document.open ();

id.document.write ( "<HTML> <HEAD>");

id.document.write ( "<BODY>");

id.document.write ( "<CENTER>");

id.document.write ( "<H2> Change text into child window. </ H2>");

id.document.write ( "<FORM NAME = f>");

id.document.write ( "<INPUT TYPE = text NAME = t SIZE = 20

MAXLENGTH = 20 VALUE = 'This is the test'> ");

id.document.write ( "<INPUT TYPE = button VALUE = 'Close the window'

onClick = window.close ()> </ FORM> ");

id.document.write ( "</ CENTER>");

id.document.write ( "</ BODY> </ HTML>");

id.document.close ();

}

<INPUT TYPE = button VALUE = "Змінити поле статусу у вікні
прикладу"

onClick = "id.defaultStatus = 'Привіт'; id.focus ();">
```

Відкриваючи вікно-нащадок, ми помістили в змінну id покажчик на об'єкт вікно id = window.open (). Тепер ми можемо використовувати її як ідентифікатор об'єкта класу Window. Використання id.focus () в нашому випадку обов'язково. При натисканні на кнопку "Змінити поле статусу у вікні приклад" відбувається передача фокусу в батьківське вікно. [9] Воно може мати розмір екрану. При цьому зміни будуть відбуватися у вікні-нащадку, яке буде приховано батьківським вікном. Для того щоб побачити зміни, треба передати фокус. Мінлива id повинна бути визначена за межами будь-яких функцій, що і зроблено. В цьому випадку він стає властивістю вікна. Якщо ми помістимо її всередині функції відкриття дочірнього вікна, то не зможемо до неї звернутися з обробника події click.

Умовна генерація HTML-розмітки на стороні браузера

Завжди приємно отримувати з сервера сторінку, налаштований під можливості нашого браузера або, більш того, під користувача. Існує тільки дві можливості генерації таких сторінок: на стороні сервера або безпосередньо у клієнта. JavaScript-код виповнюється на стороні клієнта (насправді, сервери компанії Netscape здатні виконувати JavaScript-код і на стороні сервера, тільки в цьому випадку він носить назву LiveWire-код, не плутати з LiveConnect), тому розглянемо тільки генерацію на стороні клієнта.

Для генерації HTML-розмітки контейнер SCRIPT розміщують в тілі документа. Простий приклад - вбудовування в сторінку локального часу:

<BODY>

...

<SCRIPT>

d = new Date ();

```
document.write ( "<BR>");

document.write ( "Момент завантаження сторінки:

"+ D.getHours () +": "+ d.getMinutes () +": "+ d.getSeconds ());

document.write ( "<BR>");

</ SCRIPT>

...

</ BODY>
```

1.8. Класифікація ігор

Платні ігри

Гра на фізичному носії (диски, картриджі). Класичний спосіб розповсюдження ігор, що використовувався з моменту зародження ігрової індустрії. Фізичні носії інформації - картриджі (модулі оперативної пам'яті), різні види дисків: CD, DVD, BluRay. Ігрові картриджі і диски поширюються через спеціалізовані комп'ютерні та ігрові магазини. Так само є можливість замовляти ігрові диски поштою, через інтернет-магазини.

Цифрова копія гри (продаж ігор через Інтернет). Розвиток інтернет технологій і збільшення пропускної здатності інтернет кабелів відкрило можливість скачування файлів великих обсягів по глобальній мережі. Копія гри, скачана з інтернет-ресурсу продавця, називається цифровий копією. Сам процес такого продажу називається цифровий дістрібьютіцією. Оплата за ігровий час. В деяких випадках продається не копія гри, а ігровий час. Оплата за сеанс гри використовується в іграх на аркадних ігрових автоматах. Підписка на гру використовується в деяких онлайн-іграх. Зазвичай, доступ до гри оплачується раз в місяць.

Безкоштовні ігри

Умовно безкоштовна гра (shareware). Ігри, в які можна пограти безкоштовно деякий час або несколькопервих рівнів. Для відкриття доступу до повноцінної грі, її необхідно купити. Це своєрідне розвиток ідеї демо-версії (безкоштовної пробної версії гри). Відмінність в тому, що це не частина гри, як в демо-версії, а повноцінна гра, з часів-но включеним режимом демо-версії. Такий принцип гри дуже зручний, він дозволяє гравцеві спочатку оцінити гру, а вже потім вирішувати, купувати її чи ні.

Велика частина казуальних ігор поширюється саме за таким shareware - принципом. Є такі shareware гри, де відразу ж відкритий доступ до всієї гри, але якщо за гру не сплачені гроші, то на екрані постійно по-є реклама або прохання про матеріальну допомогу авторам ігри.Бесплатная гра з мікротранзакції (free to play, free2play) (велика частина MMO-ігор). Ігри, які можна безкоштовно скачувати в інтернеті, встановлювати, грати без будь-яких обмежень. Оплату ніхто не вимагає. Але в грі є можливість покупки додаткових речей, бонусів, поліпшень, ігрової валюти за реальні гроші. При цьому, ігровий процес часто поставлений так, що витративши на гру реальні гроші, витратився гравець (Донат) отримує перевагу над звичайними гравцями. Безкоштовна гра (флеш-гра, браузерна гра, завантажується гра). Гра, за яку не потрібно платити. Зазвичай це низькоякісні гри, на які дійсно ніхто не витратить гроші. Але іноді зустрічаються гідні гри, поширювані абсолютно безкоштовно.

РОЗДІЛ 2 ПОСТАНОВА ЗАВДАННЯ І ПОРІВНЯННЯ МОЖЛИВОСТЕЙ

Ігровий жанр - група ігор, які мають схожу ігрову механіку і схожі правила гри. Існує велика кількість ігрових жанрів і кожен з них характеризується певними властивостями. Для того щоб зрозуміти до якого жанру буде ставитися будь-яка гра, потрібно «розлив-жити» гру на її складові частини і визначити їх зв'язки один з одним.

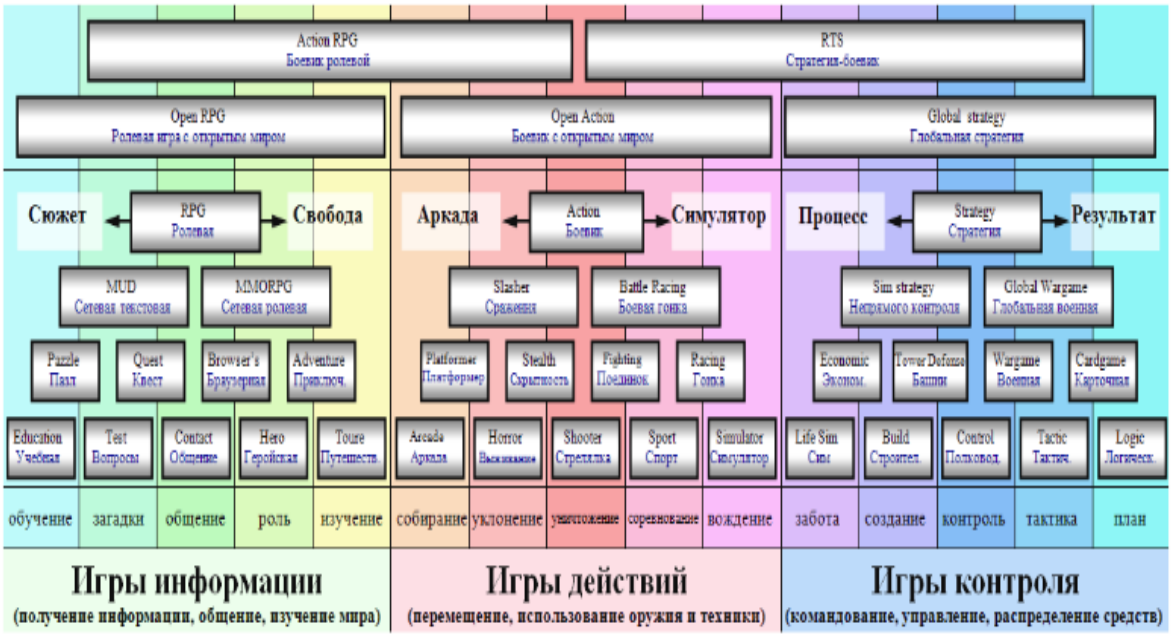

Рис. 2.1- Схема жанрів ігор

Вся схема розділена на 15 вертикальних смуг. 15 смуг розгрупувати на 3 великі групи ігор - «Ігри інформації», «Ігри дій», «Ігри контролю». У нижній частині кожної смуги вказана її сутність - базовий (неподільний) жанр, який можна назвати одним словом. Всі назви сутностей: навчання, загадки, спілкування, герой, вивчення світу, збирання, ухилення, знищення, змагання, техніка, турбота, розвиток, мікроконтроль, тактика, планування. Зверху від назви сутностей (базових ігрових механік) розташовані самі жанри у вигляді прямокутників. Найпростіші жанри перебувають внизу, вони займають всього одну смугу. Чим вище розташований жанр, тим на більшій кількості смуг він розташовується. Входження блоку жанру відразу в кілька

смуг означає, що в ньому присутня відразу кілька базових ігрових механік, об'єднаних в єдине ціле. (Наприклад: жанр «Квест» це об'єднання двох базових класів «Загадки», «Спілкування»). За цим поділом ігор виділяють 3 основних групи: рольова, бойовик і стратегія. На етапі планування імовірно, розглянувши кожен елемент схеми, майбутня гра буде ставитися до жанру «Логічна» з елементів стратегії і бойовика. Для достовірності цієї інформації розглянемо і інші поділу комп'ютерних ігор за жанрами. Внаслідок того, що критерії приналежності гри до того чи іншого жанру не визначені однозначно, класифікація комп'ютерних ігор недостатньо систематизована, і в різних джерелах дані про жанр конкретного проекту можуть розрізнятися. Існує безліч ігор, які несуть в собі якості декількох жанрів. Класифікація комп'ютерних ігор за жанром:

action:

3D-шутери, «дії-стрілялки»:

- шутери від першого і від третьої особи;

- «криваві / м'ясисті» шутери;

- тактичні шутери;

файтинг:

- побий їх усіх;

-слешер;

аркада;

2.1 Розробка проектної документації

На етапі підготовки гри розробляється концепт-документ і дизайн-документ гри. Концепт-документ (концепт) - це короткий і ємне опис

концепції (ідеї) гри, тобто, максимально стислий документ, в якому розповідається про те, якою буде гра, ніж вона буде цікава і як вона повинна виглядати після розробки. Стандартні пункти концепт документа такі [2]:

- жанр і аудиторія - вказується жанр, передбачувана аудиторія гри, наявність ліцензованого матеріалу або обмежує контенту;

- основні особливості гри (USP) - список основних особливостей гри, які гравцеві будуть цікаві і за які він викладе гроші. Найкраще, якщо в грі буде одна головна особливість;

- геймплей - найбільш повне і в той же час узагальнена (без зайвих деталей) опис всього ігрового процесу. Потрібно відповісти на питання про те, що повинен робити гравець, як він це повинен робити і, нарешті, відповісти особисто для себе на питання: «Чим гравцеві буде цікаво кожне з цих дій?» (Аналогічне питання варто поставити щодо кожної основної особливості гри);

- Сюжет - в цьому пункті варто торкнутися сюжету, але не просто передісторії подій, а саме сюжетної лінії гри. Приблизне уявлення про подальші дії створить інтригу. Величезна кількість літературного матеріалу в даному пункті нікого не зацікавить

- системні вимоги - тут потрібно оцінити, які вимоги буде пред'являти гра до «заліза» і виписати їх мінімальні і максимальні (рекомендовані) кордону. Дизайн-документ - містить в собі більш детальний опис гри, на відміну від концепт-документа. Виділяють 4 підрозділу дизайн-документа [11]:

- концепція, яка включає в себе концепт-документ гри;

- функціональний опис - це детальний опис процесу гри, з точки зору гравця;

- Опис основних технічних рішень - основа для складання наступних технічних специфікацій;

- план гри.

2.2 Порівняльна характеристика мов програмування JavaScript і PHP

Мова програмування JavaScript розроблений фірмою Netscape для створення інтерактивних HTML-документів. Це об'єктно-орієнтована мова розробки вбудованих додатків, що виконують як на стороні клієнта, так і на стороні сервера. Синтаксис мови дуже схожий на синтаксис мови Java - тому його часто називають Java-подібним. Клієнтські програми виконуються браузером перегляду Web-документів на машині користувача, серверні додатки виконуються на сервері.

При розробці обох типів додатків використовується загальний компонент мови, званий ядром і включає визначення стандартних об'єктів і конструкцій (змінні, функції, основні об'єкти і засіб LiveConnect взаємодії з Java-апплетами), і відповідні компоненти доповнень мови, містять специфічні для кожного типу додатків визначення об'єктів . Клієнтські програми безпосередньо вбудовуються в HTML-сторінки і інтерпретуються браузером у міру відображення частин документа в його вікні. Серверні додатки для збільшення продуктивності попередньо компілюються в проміжний байт-код.

Основні галузі використання мови JavaScript при створенні інтерактивних HTML-сторінок:

- Динамічне створення документа за допомогою сценарію.

- Оперативна перевірка достовірності заповнених користувачем полів форм HTML до передачі їх на сервер.

- Створення динамічних HTML-сторінок спільно з каскадними таблицями стилів і об'єктної моделлю документа.

- Взаємодія з користувачем при вирішенні "локальних" завдань, що вирішуються додатком JavaScript, вбудованому в HTML-сторінку.

Те, що стосується мови програмування PHP головним фактором при проектуванні є практичність. PHP повинен надати програмісту кошти для швидкого і ефективного вирішення поставлених завдань. Практичний характер PHP обумовлений п'ятьма важливими характеристиками:

- традиційністю,

- простотою,

- ефективністю,

- безпекою,

- гнучкістю.

Мова PHP здається знайомим програмістам, що працюють в різних областях. Багато конструкцій мови запозичені з Ci Perl, а нерідко код PHP практично не відрізняється від того, що зустрічається в типових програмах C або Pascal. Це помітно знижує початкові зусилля при вивченні PHP. Сценарій PHP може складатися з 10 000 рядків або з одного рядка - все залежить від специфіки завдання. Чи не доведеться довантажувати бібліотеки, вказувати спеціальні параметри компіляції. Механізм PHP просто починає виконувати код після першої екрануючої послідовності і продовжує виконання до того моменту, коли він зустріне парну екранує послідовність.

Якщо код має правильний синтаксис, він виконується в точності так, як вказав програміст. Ефективність є виключно важливим фактором при програмуванні для багатокористувацьких середовищ, до числа яких належить і WWW. У PHP 4.0 був реалізований механізм виділення ресурсів і забезпечена поліпшена підтримка об'єктно-орієнтованого програмування, а також засоби управління сеансом. В останній версії з'явився і механізм

підрахунку посилань (reference counting), що запобігає виділення зайвої пам'яті. PHP надає в розпорядження розробників і адміністраторів гнучкі і ефективні засоби безпеки, які умовно поділяються на дві категорії: засоби системного рівня і засоби рівня додатки.

У PHP реалізовані механізми безпеки, що знаходяться під управлінням адміністраторів; при правильному налаштуванні PHP це забезпечує максимальну свободу дій і безпеку. PHP може працювати в так званому безпечному режимі (safe mode), який обмежує можливості застосування PHP користувачами по ряду важливих показників. Наприклад, можна обмежити максимальний час виконання і використання пам'яті (неконтрольований витрата пам'яті негативно впливає на швидкодію сервера). За аналогією з cgi-bin адміністратор також може встановлювати обмеження на каталоги, в яких користувач може переглядати і виконувати сценарії PHP, а також використовувати сценарії PHP для перегляду конфіденційної інформації на сервері (наприклад, файлу passwd).

У стандартний набір функцій PHP входить ряд надійних механізмів шифрування. PHP також сумісний з багатьма додатками незалежних фірм, що дозволяє легко інтегрувати його з захищеними технологіями електронної комерції (e-commerce). Інша перевага полягає в тому, що вихідний текст сценаріїв PHP можна переглянути в браузері, оскільки сценарій компілюється до його відправки за запитом користувача. Реалізація PHP на стороні сервера запобігає викрадення нетривіальних сценаріїв користувачами, знань яких вистачає хоча б для виконання команди View Source.

Оскільки PHP є вбудованим (embedded) мовою, він відрізняється винятковою гнучкістю по відношенню до потреб розробника. Хоча PHP зазвичай рекомендується використовувати в поєднанні з HTML, він з таким же успіхом інтегрується і в JavaScript, WML, XML та інші мови. Крім того,

добре структуровані додатки PHP легко розширюються в міру необхідності (втім, це відноситься до всіх основних мов програмування). Немає проблем і з залежністю від браузерів, оскільки перед відправкою клієнту сценарії PHP повністю компілюються на стороні сервера. По суті, сценарії PHP можуть передаватися будь-яких пристроїв з браузерами, включаючи стільникові телефони, електронні записники, пейджери і портативні комп'ютери, не кажучи вже про традиційні PC.

Програмісти, що займаються допоміжними утилітами, можуть запускати PHP в режимі командного рядка. Оскільки PHP не містить коду, орієнтованого на конкретний web-сервер, користувачі не обмежуються певними серверами (можливо, незнайомими для них). Apache, Microsoft IIS, Netscape Enterprise Server, Stronghold і Zeus - PHP працює на всіх перерахованих серверах. Оскільки ці сервери працюють на різних платформах, PHP в цілому є платформенно - незалежним мовою і існує на таких платформах, як UNIX, Solaris, FreeBSD і Windows 95/98 / NT / Me / 2000 / XP. Нарешті, засоби PHP дозволяють програмісту працювати з зовнішніми компонентами, такими як Enterprise Java Beans або COM - об'єкти Win32. Завдяки цим новим можливостям PHP займає гідне місце серед сучасних технологій і забезпечує масштабування проектів до необхідних меж. Оскільки ці сервери працюють на різних платформах, PHP в цілому є платформенно - незалежним мовою і існує на таких платформах, як UNIX, Solaris, FreeBSD і Windows 95/98 / NT / Me / 2000 / XP.

2.3 Підсумкове рішення по проекту.

У даній роботі була обрано середовище програмування HTML з використанням JavaScript. Середовище програмування: Sublime Text.

Sublime Text - пропріетарний текстовий редактор.

Деякі можливості:

Швидка навігація (Goto Anything)

Командна палітра (Command Palette)

API плагінів на Python

Одночасне редагування (Split Editing)

Високий ступінь налаштування (Customize Anything)

Підтримка мов

Sublime Text підтримує , більшість мов програмування і має можливість підсвічування синтаксису для C, C ++, C #, CSS, HTML, Java, JavaScript, PHP, Python, TCL і XML.

До тих мов програмування, які включені за замовчуванням, користувачі мають можливість завантажувати плагіни для підтримки інших мов.

Менеджер пакетів

Sublime Text може бути оснащений менеджером пакетів, який дозволяє користувачеві знаходити, встановлювати, оновлювати і видаляти пакети без перезавантаження програми. Менеджер підтримує встановлені пакети в актуальному стані, завантажуючи нові версії з репозиторіїв. Крім того, він надає команди для активації і дезактивації встановлених пакетів.

Деякі особливості програми:

- Інтерфейс
- Редактор містить різні теми, з можливістю завантаження додаткових.
- Користувачі можуть побачити весь свій код в правій частині екрану у вигляді невеликої карти, при натисканні на яку можна здійснювати навігацію.

Є декілька режимів екрану. Один з них включає до 4 панелей, за допомогою яких можливо показувати від одного до чотирьох файлів

одночасно. Повноцінний (free modes) режим показує лише один файл без будь-яких додаткових меню навколо нього [5].

Виділення стовпців і множинна правка

Виділення стовпців цілком або розстановка кількох покажчиків по тексту, що робить можливим миттєву правку. Покажчики поводяться, ніби кожен з них - один в тексті. Команди типу переміщення на знак, переміщення на рядок, вибірка тексту, переміщення на слово або його частини (CamelCase, розділений дефісом або підкресленням), перехід на початок / кінець рядка та інше. Впливають на всі покажчики незалежно і відразу, дозволяючи правити складно структурований текст швидко, без використання макрокоманд або регулярних виразів.

Автодоповнення

Коли користувач набирає код, Sublime Text, в залежності від використовуваної мови, буде пропонувати різні варіанти для завершення запису. Редактор також автоматично завершує створені користувачем змінні.

Підсвічування синтаксису і висока контрастність

Темний фон Sublime Text призначений для збільшення контрастності тексту. Основні елементи синтаксису виділені різними кольорами, які краще поєднуються з темним тлом, ніж зі світлим.

Підтримка систем збірки

Sublime Text дозволяє користувачеві збирати програми і запускати їх без необхідності перемикатися на командний рядок. Користувач також може налаштувати свою систему збирання та включити автоматичну збірку програми кожного разу при збереженні коду.

Заготовки (сніппети)

Збереження фрагментів часто використовуваного коду, ключові слова для їх запуску.

Перехід по файлах

Навігаційний інструмент, який дозволяє користувачам переміщатися між файлами, а також всередині них, за допомогою нечіткого пошуку.

Інші особливості

Додатково реалізована функція автозбереження, яка допомагає користувачам не втратити виконану роботу.

Настроюються комбінації клавіш і інструмент навігації дозволяють призначати свої комбінації клавіш для меню і панелей інструментів (тільки для першої версії, у другій і третій - Command Palette).

Можливість пошуку під час набору використовується для пошуку в документі.

Функція перевірки синтаксису працює подібним же чином, перевіряючи коректність прямо під час введення.

Є можливість автоматизації за допомогою макросів і повтору останніх дій.

Команди редагування, включаючи редагування відступів, переформатування параграфів і об'єднання рядків.

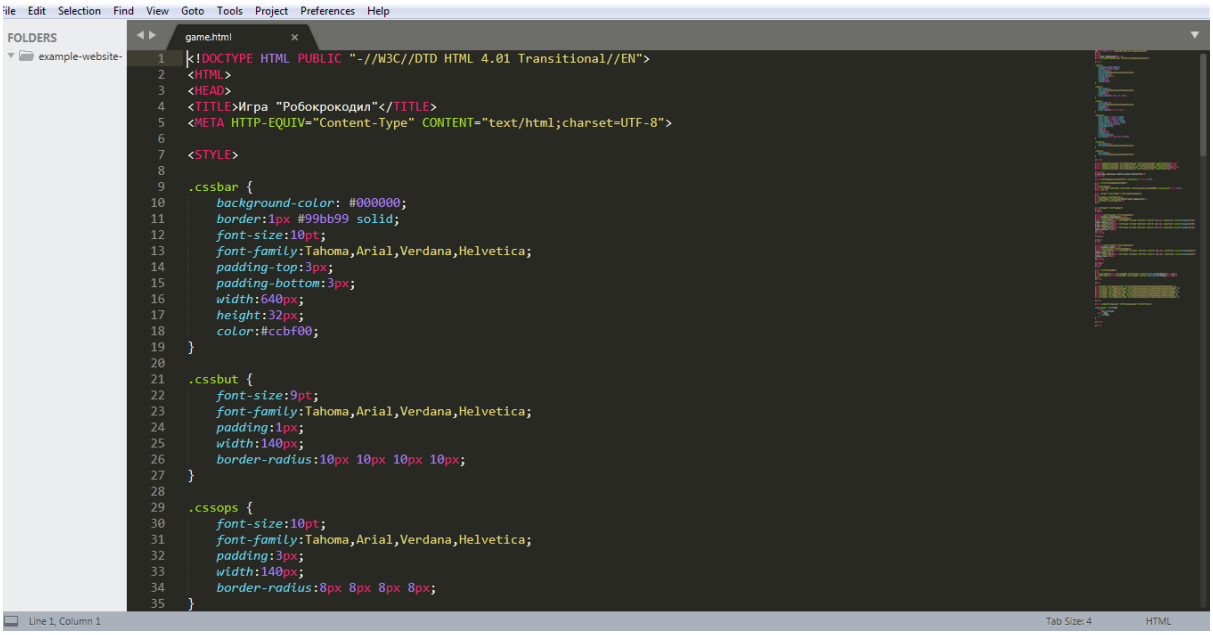

Рис. 2.2 - Зовнішній вигляд програми

РОЗДІЛ 3 РЕАЛІЗАЦІЯ

У проекті реалізувати одиночну гру із простим сюжетом.

Метою гри є ухилення від супротивників, паралельно збираючи так звані «калачики». На малюнку 3.1 зображено меню гри, де можна викликати початок гри, довідку і настройки. Всього у грі 17 рівнів.

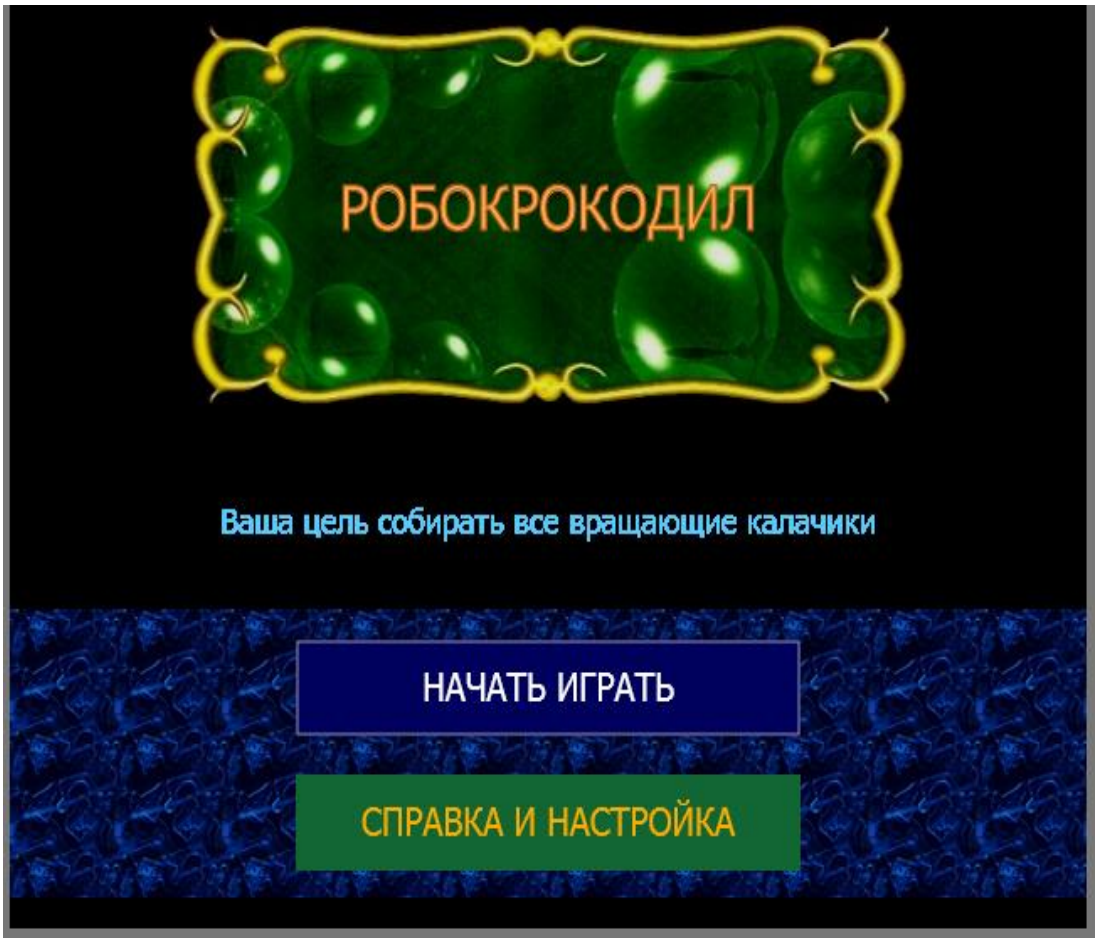

Рис. 3.1 Головне меню

У довідці гри ми можемо дізнатися все про функції управління грою (рис. 3.2).

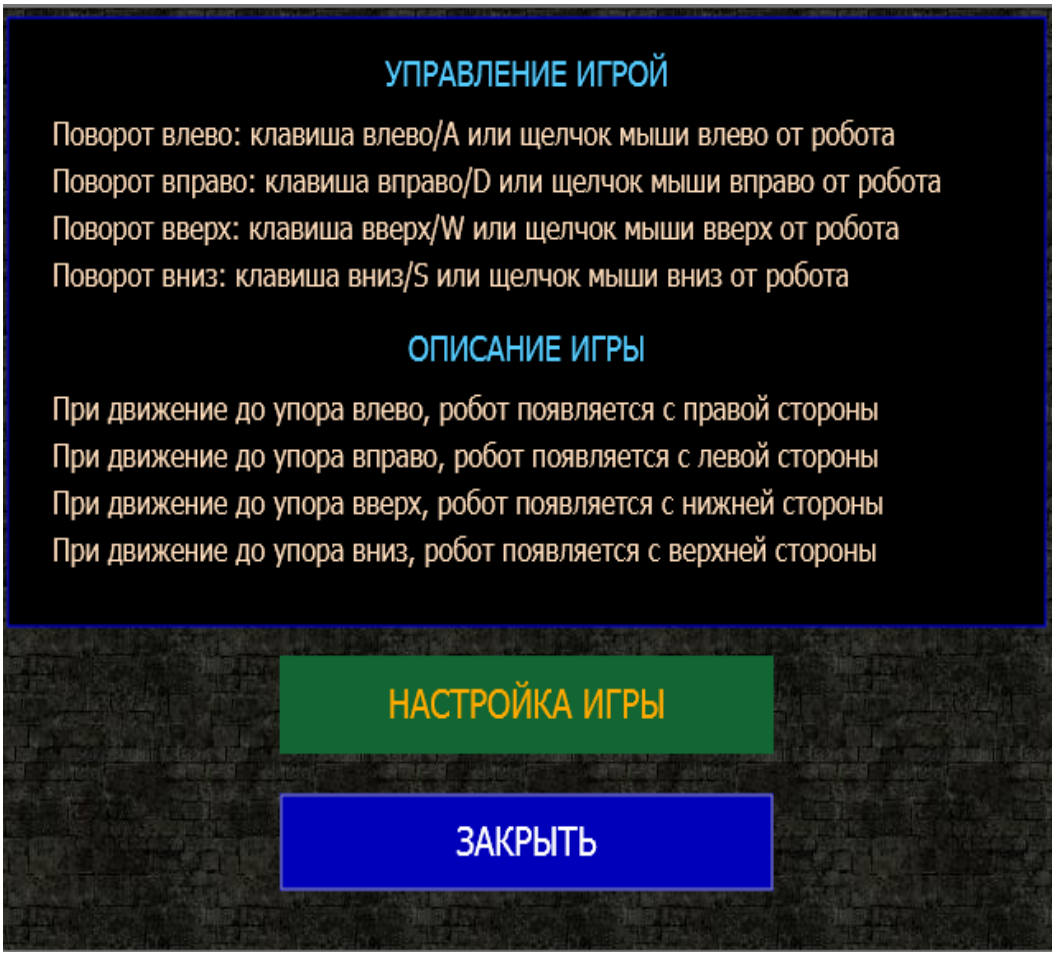

Рис. 3.2 - Довідка гри

Так само можна вибрати швидкість управління «робокрокодила» і складність гри в настройках (рис. 3.3).

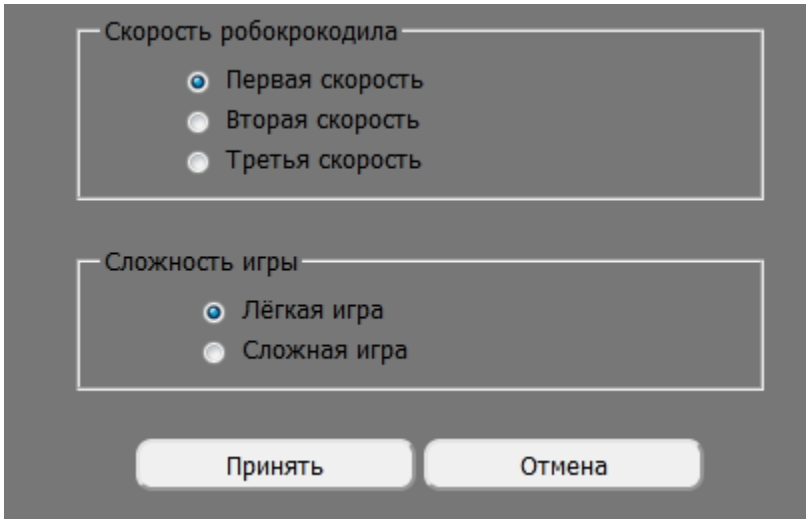

Рис. 3.3 - Налаштування гри

Далі ми маємо можливість розглянути 1 рівень гри (рис. 3.4).

Рис. 3.4 - 1 рівень гри

У разі зіткнення з противником, гравець гине і з'являється повідомлення про те, що ви програли (рис 3.5)

Рис. 3.5 - «Ви програли»

ВИСНОВКИ

Метою роботи була розробка ігрового програми на HTML із застосуванням мови JavaScript.

Були виконані завдання дослідження:

1. Проаналізувати літературу по темі дослідження та суміжних тем;
2. Розглянути основні теоретичні поняття JavaScript і HTML;
3. Розробити додаток із застосуванням JavaScript.

З кожним роком популярність комп'ютерних ігор зростає, все більше і більше людей грають в комп'ютерні ігри, вже давно перестали бути тільки розвагами для дітей і підлітків. Сучасні комп'ютерні ігри вриваються в суміжні громадські та культурні сфери - мистецтво, освіту, етику, психологію, соціальні комунікації і навіть спорт.

СПИСОК ЛІТЕРАТУРИ

1.В пошуках гарного дизайн-документа [Електронний ресурс]. - Режим доступу: http://dtf.ru/articles/read.php?id=1491

2.ДенікінА.А. Про звуці в відеоіграх // ЕНЖ «Медіамузика». No 1 (2012). [Електронний ресурс]. - режим доступу: http://mediamusic-journal.com/Issues/1_4.html.

3.Жанри комп'ютерних та відеоігор [Електронний ресурс]. - Режим доступу: http://gamesisart.ru/janr.html

4.Жанри комп'ютерних ігор (загальна схема) [Електронний ресурс]. - Режим доступу:http://gamesisart.ru/TableJanr.html

5.Жанри комп'ютерних та відеоігор. Новий і оригінальний методи поділу [Електронний ресурс]. - Режим доступу: http://gamesisart.ru/ TableJanr.html

6.Зальцман М. Комп'ютерні ігри. Як це робиться [Текст] / М.Зальцман. - М .: Логрус. РУ, 2000. - 503с.

7. Класифікація жанрів комп'ютерних ігор [Електронний ресурс] - Режим доступу: http://gamesisart.ru/janr.html

8. Класифікація комп’ютерних ігор [Електронний ресурс]. - Режим доступу: http://gamesisart.ru/game_class_all.html

9. Класифікація комп'ютерних ігор [Електронний ресурс]. - Режим доступу: https://ru.wikipedia.org/wiki/Классификация_компьютерных_игр

10.Компьютерние гри як феномен сучасної культури [Електронний ресурс] / УРФУ- Режим доступу: http://media.ls.ur-fu.ru/219/643/1374/ (

11.Основи розробки комп'ютерних ігор в XNA Game Studio. [Електронний ресурс]. - Режим доступу: http: //www.intuit.ru/studies/courses/1104/251/lecture/6445.

12.Опісаніе ігрового движка Unity3D [Електронний ресурс]. - Режим доступу: http://unity3dforge.com/

13.Программи для створення ігор [Електронний ресурс]. - Режим доступу: http://gamesisart.ru/game_dev_programms.html (дата звернення: 01.03.2016).

14.Разработка комп'ютерних ігор [Електронний ресурс]. - Режим доступу: https://ru.wikipedia.org/wiki/Разработка_компьютерных_игр

ДОДАТОК А

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
<HTML>
<HEAD>
<TITLE>Игра "Робокрокодил"</TITLE>
<META HTTP-EQUIV="Content-Type" CONTENT="text/html;charset=UTF-8">
<STYLE>
.cssbar {
	background-color: #000000;
	border:1px #99bb99 solid;
	font-size:10pt;
	font-family:Tahoma,Arial,Verdana,Helvetica;
	padding-top:3px;
	padding-bottom:3px;
	width:640px;
	height:32px;
	color:#ccbf00;
}
.cssbut {
	font-size:9pt;
	font-family:Tahoma,Arial,Verdana,Helvetica;
	padding:1px;
	width:140px;
	border-radius:10px 10px 10px 10px;
}
```

```
.cssops {
	font-size:10pt;
	font-family:Tahoma,Arial,Verdana,Helvetica;
	padding:3px;
	width:140px;
	border-radius:8px 8px 8px 8px;
}
.csspanel {
	border-left: 4px #aaccff solid;
	border-top: 4px #aaccff solid;
	border-right: 4px #77aacc solid;
	border-bottom: 4px #77aacc solid;
	background-color: #bfdfff;
	position:absolute;
	left:0px;
	top:0px;
	width:350px;
	height:270px;
	visibility:hidden;
	box-shadow:0px 0px 10px 10px #777711;
}
.cssfield {
	font-size:10pt;
	font-family:Tahoma,Arial,Verdana,Helvetica;
```

```
}

.csscopy {
        font-size:9pt;
        font-family:Tahoma,Arial,Verdana,Helvetica;
}
</STYLE>
<SCRIPT LANGUAGE="javascript" SRC="script/sdata.js"
TYPE="text/javascript" CHARSET="UTF-8"></SCRIPT>
<SCRIPT LANGUAGE="javascript" SRC="script/util.js" TYPE="text/javascript"
CHARSET="UTF-8"></SCRIPT>
<SCRIPT LANGUAGE="javascript" SRC="script/objects.js"
TYPE="text/javascript" CHARSET="UTF-8"></SCRIPT>
<SCRIPT LANGUAGE="javascript" SRC="script/game.js"
TYPE="text/javascript" CHARSET="UTF-8"></SCRIPT>

<NOSCRIPT>
<H1>Для игры, обязательно требуется наличие JavaScript !</H1>
</NOSCRIPT>

<BODY STYLE="background-color:#777777" oncontextmenu = "return false">

<TABLE STYLE="width:100%;height:100%;">
<TR>
<TD ALIGN="center">
<CANVAS ID="field" WIDTH="640" HEIGHT="480" STYLE="background-
color:#000000" oncontextmenu="return false">
```

```
</CANVAS><BR />

<TABLE ID="bar" CLASS="cssbar" STYLE="visibility:hidden">
<TR>
<TD ID="cbonus" ALIGN="left"></TD>
<TD ALIGN="center" STYLE="color:#999999">Пауза клавиша Esc</TD>
<TD ID="clevel" ALIGN="right"></TD>
</TR>


<DIV ID="cpanel" CLASS="csspanel">
<TABLE>
<TR><TD>

<FIELDSET CLASS="cssfield" STYLE="width:320px">
<LEGEND>Скорость робокрокодила</LEGEND>
<TABLE CLASS="cssfield" STYLE="width:320px">
<TR><TD ALIGN="right"><INPUT TYPE="radio" ID="vel0" NAME="vel"
VALUE="0" /></TD><TD ALIGN="left" onclick="change('vel0')">Первая
скорость</TD></TR>
<TR><TD ALIGN="right"><INPUT TYPE="radio" ID="vel1" NAME="vel"
VALUE="1" /></TD><TD ALIGN="left" onclick="change('vel1')">Вторая
скорость</TD></TR>
<TR><TD ALIGN="right"><INPUT TYPE="radio" ID="vel2" NAME="vel"
VALUE="2" /></TD><TD ALIGN="left" onclick="change('vel2')">Третья
скорость</TD></TR>
</TABLE>
```

```
</FIELDSET>


</TD></TR>


<TR><TD>

<BR />


<FIELDSET CLASS="cssfield" STYLE="width:320px">

<LEGEND>Сложность игры</LEGEND>

<TABLE CLASS="cssfield" STYLE="width:320px">

<TR><TD ALIGN="right"><INPUT TYPE="radio" ID="hard1" NAME="hrd"
VALUE="0" /></TD><TD ALIGN="left" onclick="change('hard1')">Лёгкая
игра</TD></TR>

<TR><TD ALIGN="right"><INPUT TYPE="radio" ID="hard2" NAME="hrd"
VALUE="1" /></TD><TD ALIGN="left" onclick="change('hard2')">Сложная
игра</TD></TR>

</TABLE>

</FIELDSET>


</TD></TR>

</TABLE>

<BR />


<TABLE STYLE="width:100%">

<TR>

<TD ALIGN="right"><BUTTON ID="configOK" CLASS="cssops"
onclick="config_good()">Принять</BUTTON></TD>
```

```
<TD ALIGN="left"><BUTTON ID="configNO" CLASS="cssops"
onclick="config_cancel()">Отмена</BUTTON></TD>

</TR>

</TABLE>


</DIV>


<IMG ID="user" SRC="image/user.png"
STYLE="position:absolute;left:0px;top:0px;visibility:hidden" />

<IMG ID="aliens" SRC="image/aliens.png"
STYLE="position:absolute;left:0px;top:0px;visibility:hidden" />

<IMG ID="floor" SRC="image/floor.jpg"
STYLE="position:absolute;left:0px;top:0px;visibility:hidden" />

<IMG ID="blocks" SRC="image/blocks.jpg"
STYLE="position:absolute;left:0px;top:0px;visibility:hidden" />

<IMG ID="bonus" SRC="image/bonus.png"
STYLE="position:absolute;left:0px;top:0px;visibility:hidden" />

<IMG ID="title" SRC="image/title.jpg"
STYLE="position:absolute;left:0px;top:0px;visibility:hidden" />

<IMG ID="bullet" SRC="image/bullet.png"
STYLE="position:absolute;left:0px;top:0px;visibility:hidden" />


</BODY>


<SCRIPT LANGUAGE="javascript" TYPE="text/javascript" CHARSET="UTF-
8">


window.onload = function(){
```

```
        try {
                game_create(5);
        } catch(e){
                alert(e);
        }
}

</SCRIPT>

</HTML>
```