приватне акціонерне товариство «ПРИВАТНИЙ ВИЩИЙ НАВЧАЛЬНИЙ ЗАКЛАД «ЗАПОРІЗЬКИЙ ІНСТИТУТ ЕКОНОМІКИ ТА ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ»

ДО ЗАХИСТУ ДОПУЩЕНИЙ

Зав. каф. ЕКІПЗ

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

д.е.н., проф. Левицький С.І.

**КУРСОВА РОБОТА**

З дисципліни:«Основи програмування»

ТЕМА: «Розробка гри Черв’як(змійка)»

Виконав:

ст.гр. ІПЗ-119-К9 \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ О.І. Ларіоненко

Науковий керівник:

Доцент \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Жеребцов О.А.

Запоріжжя

2021

приватне акціонерне товариство «ПРИВАТНИЙ ВИЩИЙ НАВЧАЛЬНИЙ ЗАКЛАД «ЗАПОРІЗЬКИЙ ІНСТИТУТ ЕКОНОМІКИ ТА ІНФОМАЦІЙНИХ ТЕХНОЛОГІЙ»

Кафедра економічної кібернетики та інженерії програмного забезпечення

ЗАВДАННЯ:

1. Тема роботи: Розробка гри «Черв’як» на мові програмування JavaScript , затверджена рішенням кафедри (протокол № \_\_ від \_\_\_\_\_\_\_\_\_)

2. Термін подання студентом закінченої роботи на кафедру: 25.06.2021

3. Вихідні дані до роботи (проекту) (визначаються кількісні або (та) якісні показники, яким повинен відповідати об’єкт розробки): розробка гри «Черв’як» на клієнті гравця з можливістю кастомізації.

4. Вимоги до змісту (перелік питань, що їх належить розробити) (визначаються назви розділів або (та) перелік питань, які повинні увійти до тексту роботи) :

4.1 Розгляд предметної області та огляд існуюцих аналогів;

4.2 Вибір та обґрунтування програмних засобів та технологій для реалізації проекту;

4.3 Реалізація проекту з використанням описаних методик на веб-сторінці;

4.4 Налаштування та тестування створеного програмного продукту.

5. Індивідуальний план виконання

|  |  |  |
| --- | --- | --- |
| № етапу | Зміст | Термін виконання |
| 1 | Формування теми курсового проекту | 27.04.2021 |
| 2 | Збір практичного матеріалу за темою | 04.05.2021 |
| 3 | Робота над створенням курсового проекту | 25.05.2021 |
| 4 | Оформлення пояснювальної записки та докладу | 21.06.2021 |
| 5 | Захист курсового проекту | 25.06.2021 |

Керівник (ПІБ, підпис)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

З планом ознайомлений (ПІБ, підпис)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

РЕФЕРАТ

Пояснювальна записка складається з: 35 с., 16 рисунків, 2 алгоритма, 5 лістингів, 1 схема, 8 джерел.

Об’єкт дослідження – гра «Черв’як».

Мета роботи – Розробка чату у середовищі Visual Studio code мовою JavaScript із застосуванням технології CSS та HTML.

Методи дослідження – використання технологій CSS, HTML,worm1.3.js, Visual Studio code.

Розглянуто методи реалізації гри на мові JavaScript. Зокрема, розглянуто застосування різних методів оформлення та взаємодії з грою.

Проводиться аналіз засобів гри з використанням HTML сторінок. Обґрунтовується вибір технології та середовища розробки гри. Розроблено візуаліний інтерфейс взаємодії користувача з кодом гри.

JS-GAME, CSS, HTML, SNAKE.

ЗМІСТ

[ВСТУП 6](#_Toc75438403)

[РОЗДІЛ 1 МОБІЛЬНІ ІГРИ 7](#_Toc75438404)

[1.1 Історія виникнення мобільних ігор 7](#_Toc75438405)

[1.2 Ринок відео ігор 8](#_Toc75438406)

[1.3 Аналоги гри «Черв’як» 9](#_Toc75438407)

[1.4 Висновки за розділом 12](#_Toc75438408)

[РОЗДІЛ 2 ОГЛЯД ПРОГРАМНОГО РІШЕННЯ 13](#_Toc75438409)

[2.1 Мова програмування JavaScript 13](#_Toc75438410)

[2.2 Візуальна середа розробки 14](#_Toc75438411)

[2.3 Висновки за розділом 15](#_Toc75438412)

[РОЗДІЛ 3 ПРОГРАМНИЙ МОДУЛЬ 16](#_Toc75438413)

[3.1 Реалізація гри 16](#_Toc75438414)

[3.2 Технологічний комплекс 19](#_Toc75438415)

[3.3 Організація середовища для розробки чату 20](#_Toc75438416)

[3.4 Структура додатку 20](#_Toc75438417)

[3.5 Реалізація механік гри 22](#_Toc75438418)

[3.6 Візуальне оформлення та інтерфейс 24](#_Toc75438419)

[3.7 Висновки за розділом 27](#_Toc75438420)

[ВИСНОВКИ 28](#_Toc75438421)

[ПЕРЕЛІК ПОСИЛАНЬ 30](#_Toc75438422)

[ДОДАТКИ 31](#_Toc75438423)

[ДОДАТОК А 32](#_Toc75438424)

# ВСТУП

В світі існує безліч різноманітних ігор, що розраховані на велику кількість користувачів. Це можуть бути головоломки, шутери, платформери, класичні та інші. Всі ці ігри розробляють великі ігрові студії на різних мовах програмування наприклад: C++, C#, Java тощо. Прикладом гри, побудованої на цих мовах , є прикладна програма або додаток — гра Змійка. У цій роботі я буду створювати таку гру за допомогою JavaScript та HTML в купі з CSS.

Метою даної курсової роботи є збір теоретичних відомостей про сучасні ігрові тенденції та ринок ігор взагалі; огляд та аналіз розробки ігор різними мовами програмування; визначення оптимальних платформ для гри, спираючись на аналіз процесу розробки гри «Черв’як».

Для повноцінного додатку потрібно розробити оформлення інтерфейсу сторінки та взаємодію на веб-сторінці з кодом гри, обрати метод взаємодії гравця на код гри, налаштувати взаємодію між гравцем та грою. Для цього потрібно обрати мову програмування, середовище розробки, тощо.

В ході розробки будуть вивчені різноманітні допоміжні методи в розробці гри та реалізовані наступні кроки:

* обдумати та створити робочий прототип;
* створити моделі або схеми об’єктів додатку;
* реалізувати збереження інформації на клієнтській частині;
* налаштувати клієнтську взаємодію з грою;
* оглянути гру та зробити висновки.

# РОЗДІЛ 1 МОБІЛЬНІ ІГРИ

## 1.1 Історія виникнення мобільних ігор

Виникненню відеоігор передував розвиток програмованих комп'ютерів для військових цілей. Їх пізнішому виникненню як виду розваг сприяв розвиток телебачення. Хоча воно існувало ще до Другої світової війни, конфлікт змусив виробників відмовитися від виробництва телевізорів на користь дисплеїв для радарів і іншого устаткування для збройних сил. Виникла необхідність в нових технологіях і методах управління складними комплексними проектами розробки програмних систем. Але не кожен міг собі позволити комп’ютер або ігрову приставку яка могла задовільнити запити гравців. Недоліком цих комп’ютерів та ігрових приставок став їх розмір та скрутна можливість транспортування для гри в будь-якому місці.

В кінці XX століття почала зароджуватися мобільна-ігрова індустрія. У 1994 році компанія Siemens представила перший в світі GSM-телефон Siemens S1 MARATHON. Якщо на цьому телефоні ввести спеціальний код, то можна було запустити гру Тетріс.

Трохи згодом, у 1997 році Nokia 6110 став першим телефоном з вбудованою грою Snake, яка запускалася з меню. У грі потрібно було поглинати чорні точки на екрані. Тим більше змійка поглинала точок, тим довше вона ставала. Гра відразу знайшла безліч фанатів, а сама Nokia впроваджувала її майже на всі свої телефони.

Індустрія телефонних ігор зробила для себе великий крок у 2001 році,саме тоді з’явилася нова платформа Java2ME, тепер користувачі могли самостійно завантажувати ігри з інтернету або з інших джерел. З'явилися безліч сайтів з Java-іграми на будь-який жанр. Великі студії як EA, Ubisoft, Konami, Disney випустили мобільні версії таких, як серія Assassin's Creed, The Sims, Need For Speed, PES, Command & Conquer і інші популярні ігри.

У 2007-2008 році почалася нова ера правління Android та IOS, з появою операційних систем Android і iOS термін «телефон» був замінений на «смартфон». З кожним роком смартфони ставали значно потужніше, відкривши нові можливості в створенні ігор для розробників. Першою популярною грою для обох операційних систем став Angry Birds. Через рік релізу її скачали більше мільйона разів.

## 1.2 Ринок відео ігор

Наразі індустрія відеоігор є лідируючим сегментом в індустрії розваг по всьому світу. Число геймерів зростає (3,1 млрд. гравців у 2020 р.), а сам геймінг стає високооплачуваною професією. Формується потужна екосистема. Пізніше серед дослідників утвердилася думка, що 1950-і стали точкою неповернення для відеоігор.

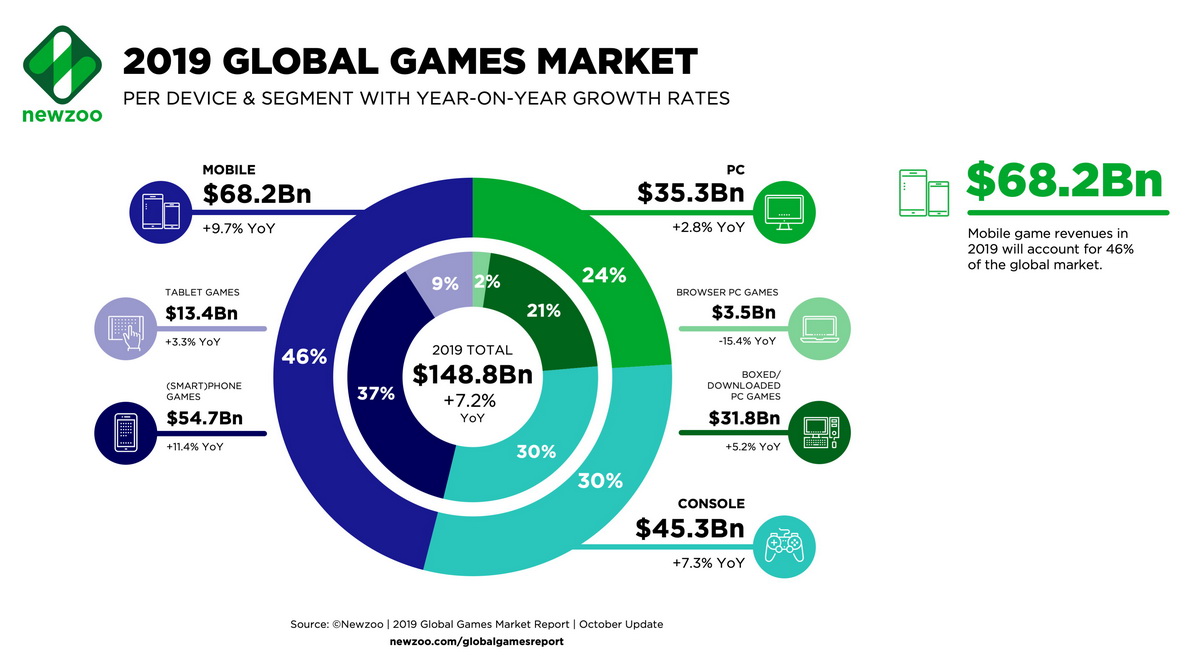


Рисунок 1.1 – діаграма капіталізації ринку ігрових розваг

У 2019 р. світовий ігровий ринок зросте до $148,8 млрд. Про це йдеться у звіті про ігрову індустрію у світі та в окремих регіонах Global Games Market Report аналітичної компанії Newzoo . Одним із основних чинників, що сприяють цьому перегляду, є уповільнення витрат на сегмент ігрових приставок. За оцінками компанії, сегмент ринку консольних ігор досягне $45,3 млрд у цьому році (+7,3% в річному обчисленні) — в порівнянні з попередньою оцінкою в $47,9 млрд.

З діаграми капіталізації ринку ігрових розваг можна зробити висновок що індустрія мобільних розваг по обсязі капіталізації є лідером що робить її грошовою жилкою для розробників.

## 1.3 Аналоги гри «Черв’як»

Гра «Черв’як» - це та ж сама гра змійка але з іншою назвою. При розробці гри я спирався на вже існуючі аналоги беручи тільки цікаві на моє бачення фішки яких не має в інших аналогаї.

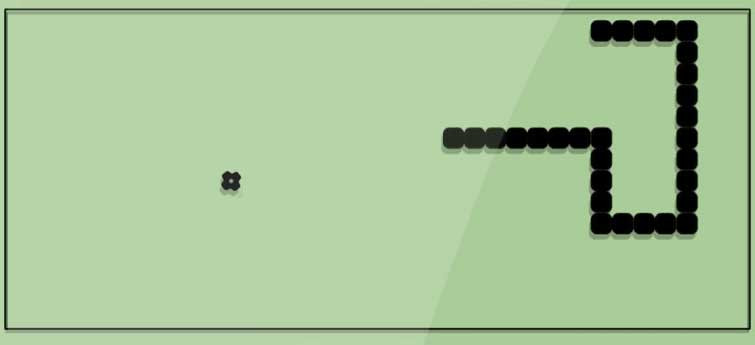


Рисунок 1.2—зовнішній вигляд оригинальної «Змейка» (Snake) от Nokia 1997 року.

На прикладі візуального оформлення змійки 1997 року було вирішено взяти квадратну форму змійки для спрощення розрахунку переміщення «Черв’яка» по ігровому полю.



Рисунок 1.3—зовнішній вигляд черв’яка у грі «Черв’як» .

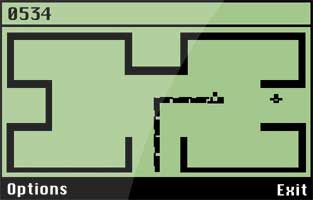


Рисунок 1.4— зовнішній вигляд продовження гри змійка «Snake 2» от Nokia 1997 року.

На прикладі візуального оформлення продовження «Snake 2» було вирішено реалізувати лічильник та можливість налаштування з подальшою реалізацією можливості зміни кольору фону, кольору хвоста, їжі.

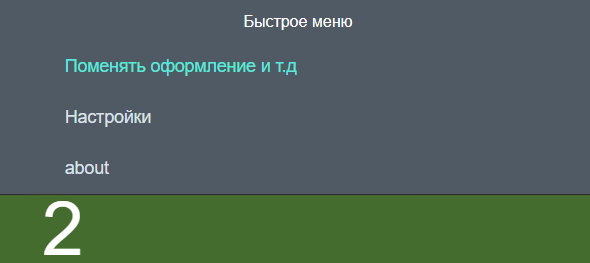


Рисунок 1.5—реалізація та зовнішній вигляд меню взаємодії та сам лічильник у розробленій грі «Черв’як» .

Рисунок 1.6—зовнішній вигляд меню «оформлення» у розробленій грі «Черв’як»



Рисунок 1.7—зовнішній вигляд меню «налаштування» у розробленій грі «Черв’як».

## 1.4 Висновки за розділом

Проаналізувавши історію мобільної ігрової індутрії та ринок капіталізації відео ігор по платформам я дійшов до висновку що амбіціозною платформою реалізації моєї гри є мобільний сегмент та персональні комп’ютери але було вирішено на певний період часу обмежитися платформою персональних компьютерів. Зібравши інформацію про аналоги розробленої гри «Черв’як» було відібрано тільки потрібні та добре себе зарекомендовані часом фішки.

З цього всього було прийнято рішення о розборці гри «Черв’як» на мові JavaScript з використанням технологій CSS та HTML.

# РОЗДІЛ 2 ОГЛЯД ПРОГРАМНОГО РІШЕННЯ

## 2.1 Мова програмування JavaScript

JavaScript (JS) — динамічна, об'єктно-орієнтована прототипна мова програмування. Реалізація стандарту ECMAScript. Найчастіше використовується для створення сценаріїв веб-сторінок, що надає можливість на стороні клієнта (пристрої кінцевого користувача) взаємодіяти з користувачем, керувати браузером, асинхронно обмінюватися даними з сервером, змінювати структуру та зовнішній вигляд веб-сторінки.

JavaScript класифікують як прототипну (підмножина об'єктно-орієнтованої), скриптову мову програмування з динамічною типізацією. Окрім прототипної, JavaScript також частково підтримує інші парадигми програмування—імперативну та частково функціональнуі деякі відповідні архітектурні властивості, зокрема: динамічна та слабка типізація, автоматичне керування пам'яттю, прототипне наслідування, функції як об'єкти першого класу.

Мова JavaScript використовується для:

* написання сценаріїв веб-сторінок для надання їм інтерактивності;
* створення односторінкових веб-застосунків (React, AngularJS, Vue.js);
* програмування на стороні сервера (Node.js);
* стаціонарних застосунків (Electron, NW.js);
* мобільних застосунків (ReactNative, Cordova);
* сценаріїв в прикладному програмному забезпечені (ПЗ), наприклад, в програмах зі складу AdobeCreativeSuite чи ApacheJMeter;
* всередині PDF-документів тощо.

JavaScript, наразі, є однією з найпопулярніших мов програмування в Iнтернеті. Але спочатку багато професійних програмістів скептично ставилися до мови, цільова аудиторія якої складалася з програмістів-любителів. Поява AJAX змінила ситуацію та повернула увагу професійної спільноти до мови, а подальші модифікації мови за стандартами ES2015 та ES2017 внесли багато корисних можливостей, яких не вистачало для ефективного програмування. В результаті, були розроблені та покращені багато практик використання JavaScript (зокрема, тестування та налагодження), створені бібліотеки та фреймворки, поширилося використання JavaScript поза браузером.

## 2.2 Візуальна середа розробки

Visual Studio Code - це один з найбільш популярних редакторів коду, розроблений корпорацією Microsoft. Він поширюється в безкоштовному доступі і підтримується всіма актуальними операційними системами: Windows, Linux і macOS. VS Code є звичайним текстовим редактор з можливістю підключення різних плагінів, що дає можливість працювати з різними мовами програмування для розробки будь-якого IT-продукту.

Visual Studio code підтримує мови JavaScript, Python, CSS, HTML, Java та інші. Він забезпечує автодоповнення, аналіз коду «на льоту», навігацію по коду, рефакторинг, інтеграцію з системами управління версіями. Важливою перевагою інтегрованого середовища розробки є встроєний додаток де користувач може завантажити плагіни та інше для Visual studio code не виходячи з програми, у тому числі, редактор містить вбудований зневаджувач, інструменти для роботи з Git, автодоповнення типових конструкцій і контекстної підказки. Продукт підтримує розробку для платформ ASP.NET і Node.js, і позиціюється як легковагове рішення, що дозволяє обійтися без повного інтегрованого середовища розробки.

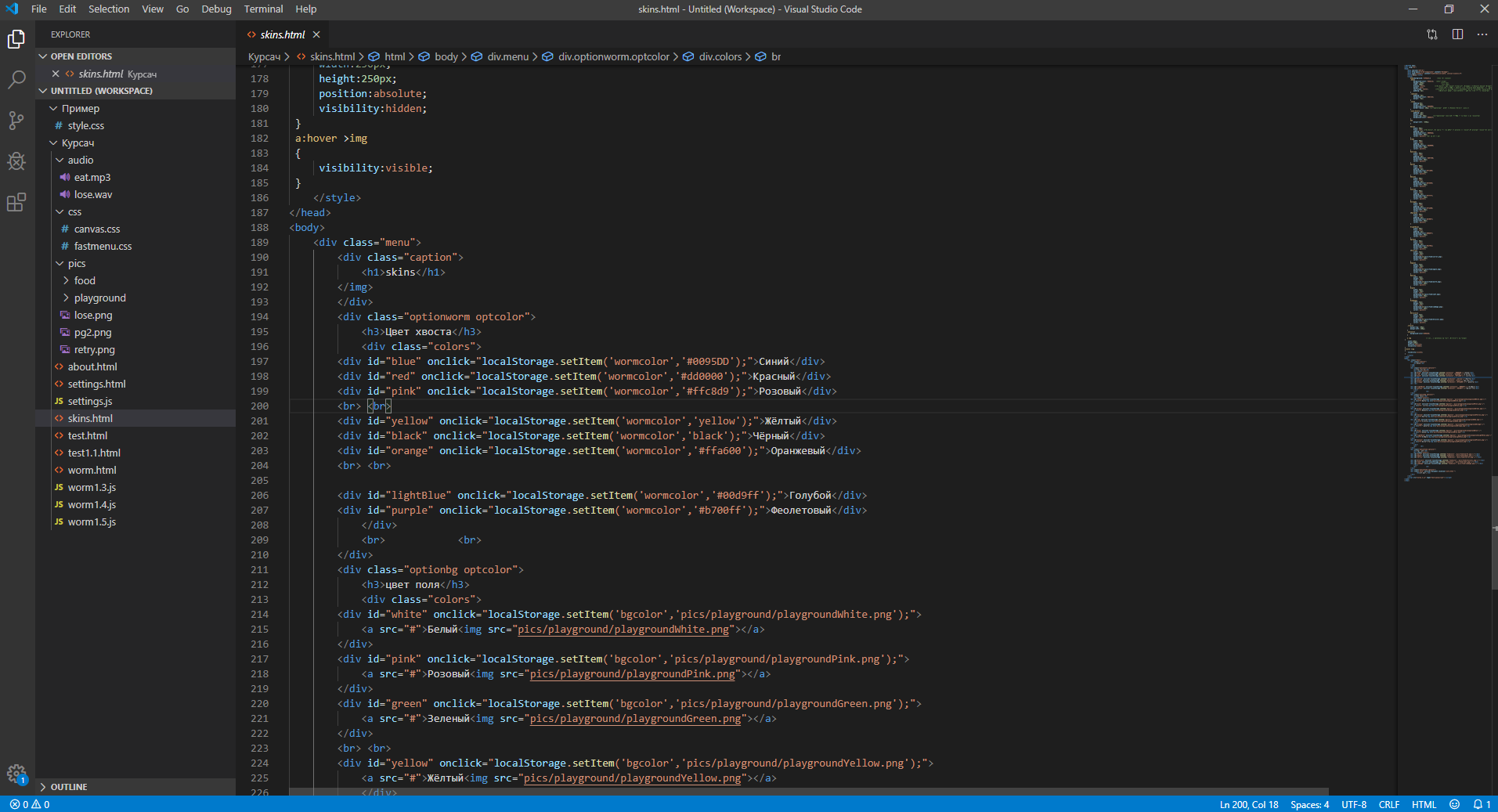


Рисунок 2.8— Інтерфейс програми Visual Studio code

## 2.3 Висновки за розділом

Розглянувши засоби програмування, було прийнято рішення створювати проект за допомогою JavaScript у середовищі Visual Studio code.

# РОЗДІЛ 3 ПРОГРАМНИЙ МОДУЛЬ

Щоб створити гру «Черв'як», потрібно розробити код програми, візуальне оформлення та реалізуати гру на веб-сторінці браузера. Завдяки Visual Studio code і можливістю завантаження плагінів, це завдання спрощується. Також необхідна веб-сторінка де буде сама гра, де будуть можливості переміщення між веб-сторінками які будуть виконувати роль кімнат де користувач буде налаштовувати гру під себе.

## 3.1 Реалізація гри

Для початку потрібно розділити розробку програми на 2 частини: сам код гри та оформлення веб-сторінки. Потрібно буде працювати з цими частинами, щоб зробити задумане. Взаємодія між кодом гри та веб сторінкою показана на рисунку 3.1.

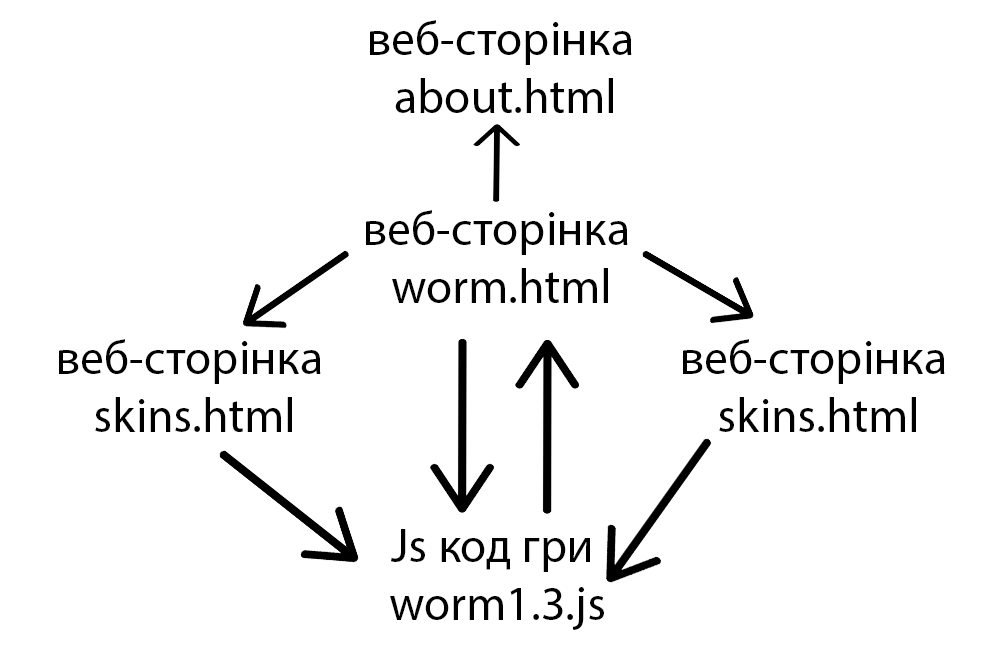
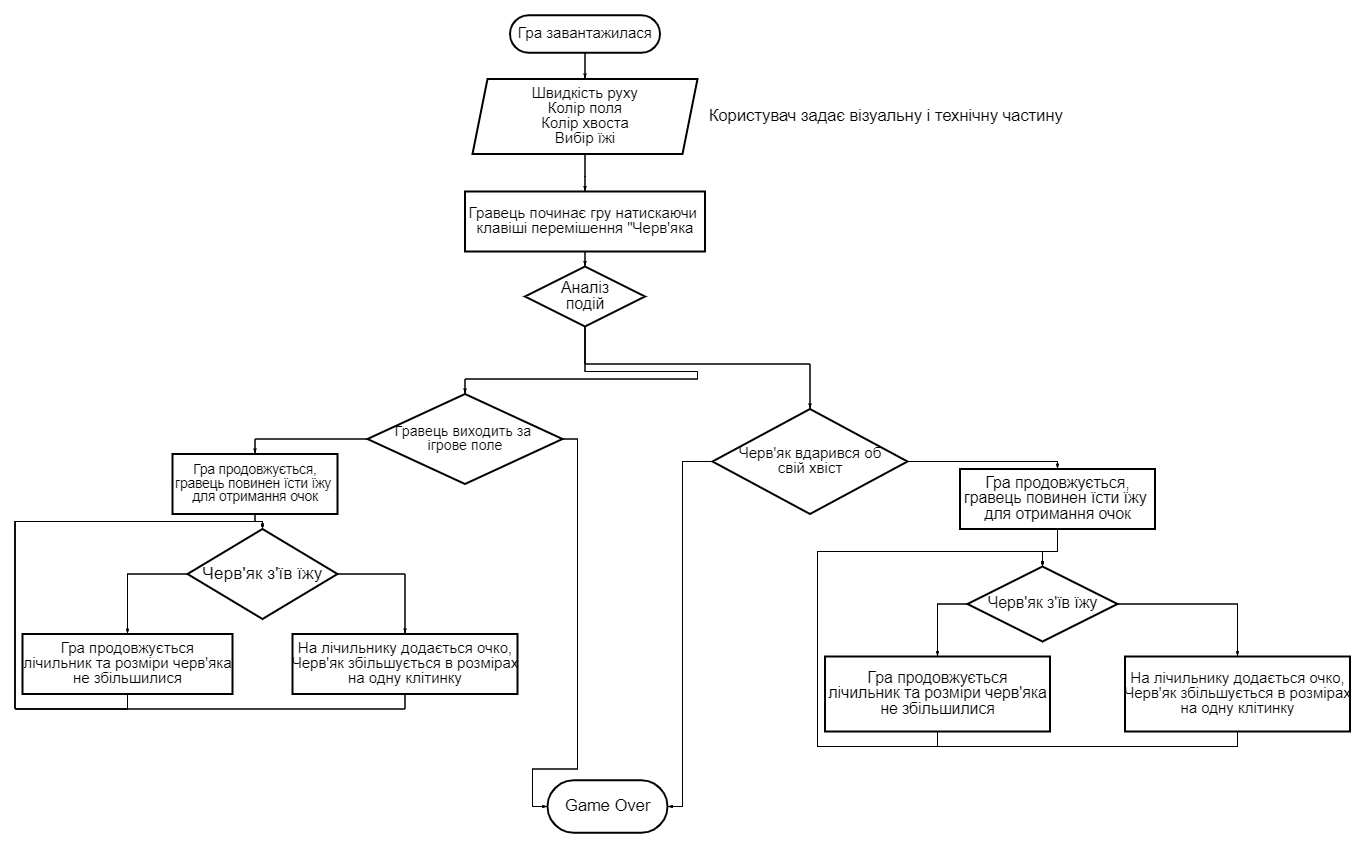


Схема 3.1 — Взаємодія між основними файлами проекту



Алгоритм 3.1— Алгоритм роботи гри

Алгоритм 3.1 зображує алгоритм роботи гри. Користувач виступає в ролі відправника команд вхідні данні яких обробляються програмним кодом. Більш детально його можна роздивитися в додатку B.



Алгоритм 3.2— Алгоритм взаємодії гравця на гру через skins.html

## 3.2 Технологічний комплекс

Під час створення гри були використані наступні інструменти:

* для створення спрайтів :Adobe Photoshop CC2017, Paint, Adobe illustrator;
* для оформлення зовнішнього вигляду:CSS.

На рисунку 3.1 приклад розбиття сторінки на компоненти, де кожний виділений фрагмент сторінки вважається компонентом.



Рисунок 3.9— складові проекту

## 3.3 Організація середовища для розробки чату

По-перше, потрібно організувати середу розробки гри. Потрібно створити сам скрипт, організувати веб-сторінки для взаємодії з скриптом, організувати структуру файлів для зручного орієнтування та взаємодії з файловою структурою гри.

## 3.4 Структура додатку

Розглянемо поступово структуру файлів гри на рисунку 3.2:

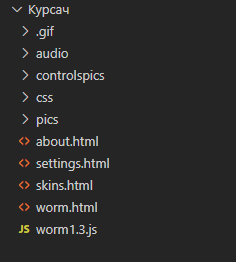


Рисунок 3.10— Структура файлів гри

Папка Audio **— папка зі** звуками які були приєднані до гри для звукової візуалізації конкретної події у грі.

Папка CSS**—** папка з налаштуваннями та візуальними оформленнями.

Папка Pics**—** ця папка має під папки в яких класифікується картинки їжі, ігрового поля також в цій папці є файли без папкової класифікації такі як:

* Lose.png-картинка яка з’являється при програші і містить текст «Game Over»;



Рисунок 3.11— Lose.png

* Pg2.png-картинка яка відповідає за задній фон гри;
* Retry.png-картинка з пропозицією повторити спробу.

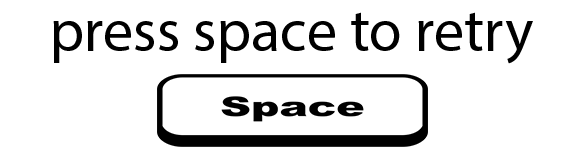


Рисунок 3.12— Retry.png

Під папка playground**— має в собі варіації ігрового поля в форматі png.**

Під папка food**— має в собі варіації їжі в форматі png.**

Папка .gif **— папка з зацикленими відео.**

Папка controlpics **— папка з картинками клавіш взаємодії.**

About.html**—веб-сторінка де описано хто написав цю курсову роботу.**

Skins.html**—веб-сторінка на якій користувач обирає оформлення гри за допомогою цієї веб-сторінки яка впливає і змінює данні в коді гри.**

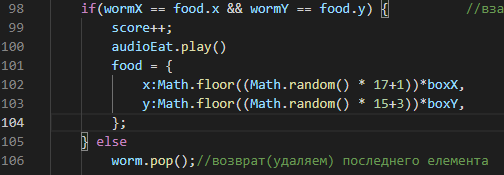
Settings.html**—веб-сторінка на якій користувач налаштовує гру під себе за допомогою цієї веб-сторінки яка впливає і змінює данні в коді гри.**

Worm1.3.js **—**код самої гри.

Worm.html**—**веб-сторінка на якій проходить сама гра в гру «Черв’як» та інше.

## **3.5** Реалізація механік гри

Потрібно створити створити реалізацію взаємодії черв'яка з їжею, прикладом цього є зображення лістингу 3.1.

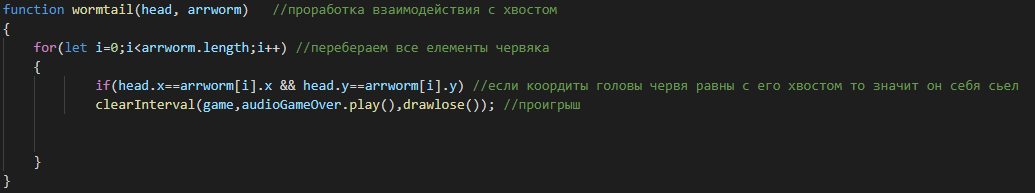


Лістинг 3.1— реалізація взаємодії черв’яка з їжею

В Лістингу 3.1 наведено спосіб реалізації через if() else де ми передаємо в if порівняння координат їжі з координатами черв’яка, після чого якщо координати збігаються йде послідовність подій: в лічильник передається інформація про збільшення на одне очко після чого йде випадкова поява їжі на ігровому полі за допомогою Math.random() помноженої на кількість клітинок по осі X та осі Y до яких додано відступ в клітинках по краям, це все ми округляємо і множимо на розміри однієї клітинки грального поля ,інакше йде видалення останнього елементу масива якщо черв’як не з`їв їжу але якщо він з’їв то ми не видаляємо останній елемент масиву а додаємо новий Лістинг 3.2 після чого черв’як збільшується в розмірах.

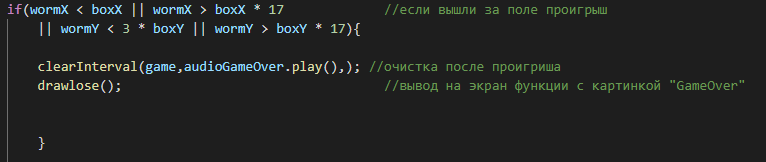
https://i.imgur.com/TrU39ZS.png

Лістинг 3.2— реалізація додавання елементу



Лістинг 3.3— реалізація зіткнення з хвостом

Для реалізації зіткнення з хвостом дивитися Лістинг 3.3 створюємо функцію в яку передаємо head перший елемент черв’яка який виступає головою, arrworm масив який формує хвіст. Робимо цикл де перебираємо всі елементи черв’яка в якому робимо порівняння якщо координати голови рівні з координатами масиву елемента хвоста.



Лістинг 3.4— реалізація зіткнення з хвостом

Реалізації виходу за поле дивитися Лістинг 3.4 робимо передаємо в порівняння якщо координати черв’яка по осі X менша за кількість пікселів квадрата поля по осі X або координати черв’яка по осі X більші за кількість пікселів квадрата поля помножених на кількість квадратів по осі Х 17 квадратів або координати черв’яка по осі Y менша за 3 помноженої на кількість пікселів квадрата поля по осі Y або координати черв’яка по осі Y більші за координати черв’яка по осі X більші за кількість пікселів квадрата поля помножених на кількість квадратів по осі Х 17 квадратів.

……………………………………………………………………………….

## 3.6 Візуальне оформлення та інтерфейс

Як відомо, JavaScript, CSS та HTML друзі не розлий вода. Моя робота не виняток. На лістингу 3.5**—**файл worm.html, який і є веб-сторінкою на якій все відбувається.

 Лістинг 3.5—код «worm.html»

На рисунку 3.13 зображено меню швидкого доступу до інших веб-сторінок. Зовнішній вигляд меню має дуже мінімалістичне оформлення і не має родзинок, єдине що я можу додати що реалізовно підсвічення надпису при наведенні курсора на текст приклад рисунку 3.33.

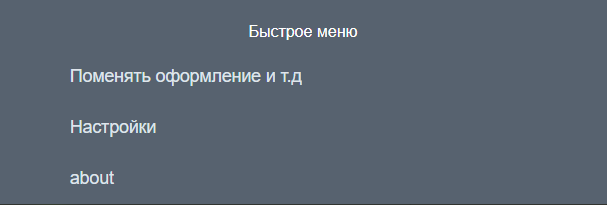


Рисунок 3.13— fastMenu

Змінити оформлення та інше**—** це кнопка, яка відкриває html сторінку в якій користувачеві пропонується змінити оформлення і візуальну частину гри.

Для наочності розглянемо html сторінку на якій розробимо меню можливої кастомізації гри «Черв’як»,на рисунку 3.14 можна побачити вигляд меню пропозицій по можливій кастомізації гри.

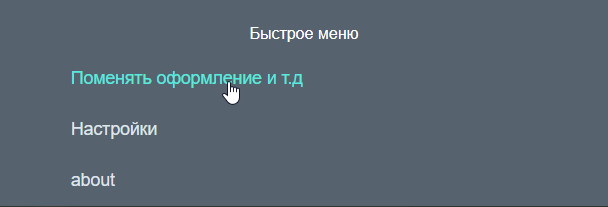


Рисунок 3.14— приклад підсвічування тексту



Рисунок 3.15— Меню можливої кастомізації гри «Черв’як»



Рисунок 3.16— Меню налаштування гри «Черв’як»

## 3.7 Висновки за розділом

Було створено гра «Черв’як» актуальність якої заключалася в траті вільного часу на розваги з метею розслаблення та тренерування моторики пальців та розвитку реакції та швидкого прийняття рішень.

Розроблена гра є кросплатформенною майже на всіх браузерах, тестування крос платформи було проведено в браузерах:

Chrome;

Mozilla Firefox;

Opera.

# ВИСНОВКИ

Під час аналізу доступних джерел було проведено дослідження сферу мобільних ігор, під час якого була проведено дослідження про актуальність платформи розробки.  Додатково було складено алгоритм роботи гри. Були проаналізовані популярні засоби розробки. При аналізі існуючих розробок, було проведено їх порівняння і виділені їхні переваги і недоліки. В ході аналізу стало ясно, що при розробці гри варто звернути увагу на взаємодію веб-сторінок між собою та кодом гри. Це потрібно для того, щоб користувач зручно міг через веб-інтерфейс взаємодіяти на гру це збільшує зручність гри.

Ґрунтуючись на отриманій в ході дослідження інформації, було вирішено розробити гру на мові програмування JavaScript з використанням технологій CSS та HTML. Таке рішення було прийнято з двох причин:

Веб-сокетишвидші і менш навантажують браузери.

HTML- це зручний спосіб виведення будь-якої інформації на веб-сторінку для зручної обробки інформації користувачем.

СSS- це зручний засіб візульного оформлення веб-сторінок, потрібна технологія для оформлення красивого інтерфейсу програми.

Після вибору засобів розробки було розпочато вивчення CSS, а також розробка самого проекту. В ході розробки було та отримані наступні знання та вміння:

* створення візульної частини веб-сторінки з використанням технології CSS;
* збереження інформації у браузері;
* створення і написання кнопок взаємодії з html файлу на js;
* налаштування та тестування розробленої програми;
* компіляція проекту.

Освоєння CSS та HTML несе важливий характер, так як допоки інтернет живий ці технології будуть затребуванні.Таким чином, в ході реалізації проекту були виконані наступні завдання:

* вивчено історію мобільниї ігор.
* обрано жанр, вид та платформа для гри;
* розроблено сценарій і концепція основних елементів;
* вибрано і вивчено засоби реалізації;
* реалізовано прототип гри.

Завдяки розширенню в середовищі розробки проекту Visual Studio code

«Live Server» було реалізовано запуск гри на клієнті, тобто запуск локального сервера розробки з функцією перезавантаження в реальному часі для статичних і динамічних сторінок. На момент розробки та створення робочої версії гри функціоналу цього розширення було достатньо.

Якщо розвиток гри продовжиться тоді треба буде приглядатися до якогось сервер хосту для розміщення гри та подальший її пошук в адресній строчці браузера а не запуск через локальний сервер.

# ПЕРЕЛІК ПОСИЛАНЬ

1. Ринок комп’ютерних ігор: https://investory.news/rinok-kompyuternix-igor-zris-do-majzhe-150-milyardiv/
2. Справочник CSS: http://htmlbook.ru/css
3. Справочник по HTML: http://htmlbook.ru/html
4. YouTube канал

WebForMySelf Урок 5. Javascript. Практика для начинающих. Изменение картинки: https://youtu.be/EatnlC6pofY?list=LL

1. Справочник по JavaScript:https://developer.mozilla.org/ru/docs/Web/JavaScript/Reference
2. Історія розвитку мобільних ігор: <https://trashbox.ru/topics/109325/istoriya-razvitiya-mobilnyh-igr>.
3. Історія гри «Змійка»: <https://habr.com/ru/company/microsoftlumia/blog/136629/>.
4. Visual Studio code : https://code.visualstudio.com/.

# ДОДАТКИ

# ДОДАТОК А

Вихідний код гри «Черв’як» worm1.3.js

const canvas = document.getElementById("game"); //подтягиваем canvas в наш код через id "game"

const ctx = canvas.getContext("2d"); //тут указываем какой будет игра, 2 мерной

const pg = new Image(); //игровой фон

pg.rc = "pics/pg2.png";

const foodpic = new Image();//картинку еды

foodpic.src = "pics/food2.png"

const lose = new Image(); //экран проиграша

lose.src="pics/lose.png";

let playing =true; //

let playground = new Image(); //игровое поле

playground.src = "pics/playgroundGreen.png"

let audioEat = new Audio("audio/eat.mp3") //звук употребления еды

let audioGameOver = new Audio('audio/lose.wav') //звук проигрыша

let retry=new Image();//картинка попробывать снова

retry.src="pics/retry.png";

let boxX =32; //ширина

let boxY =32;//высота

let score = 0;//стартовое количество очков

let food={ //рандом спавн еды

x:Math.floor((Math.random() \* 17+1))\*boxX, //используем рандом с диапазоном до 17 по x,отступ 1 boxX округляем до целого и умножаем на размер квадрата по X

y:Math.floor((Math.random() \* 15+3))\*boxY,//используем рандом с диапазоном до 15 по y,отступ 3 boxY округляем до целого и умножаем на размер квадрата по Y

};

let worm=[]; // создаем пустой массив данных

worm[0]={ //место появления червяка(ровно по центру)

x:9\*boxX,

y:10\*boxY,

function drawlose(){ //функция появления надписи game over ,retry

ctx.drawImage(lose, 0, 0);

ctx.drawImage(retry, 0, 0);

}

document.addEventListener("keydown",direction); //оброботчкик событий который будет обротатывать любые запросы на странице(когда клавиша была нажата)

let bob; //переменная для помощи в управлении (в названии нет смысла)

function direction(event) { //управление червяком с помощью стрелок

if(event.keyCode == localStorage.getItem('controlLeft') && bob != "right") //добавлено что нельзя нажать стрелку передвежения вправо пока червяк ползет влево

bob = "left";

else if(event.keyCode == localStorage.getItem('controlUp') && bob != "down") //добавлено что нельзя нажать стрелку передвежения вниз пока червяк ползет вверх

bob = "up";

else if(event.keyCode == localStorage.getItem('controlRight') && bob != "left") //добавлено что нельзя нажать стрелку передвежения влево пока червяк ползет вправо

bob = "right";

else if(event.keyCode == localStorage.getItem('controlDown') && bob != "up") //добавлено что нельзя нажать стрелку передвежения вверх пока червяк ползет вниз

bob = "down";

}

function wormtail(head, arrworm) //проработка взаимодействия с хвостом

{

for(let i=0;i<arrworm.length;i++) //перебераем все елементы червяка

{

if(head.x==arrworm[i].x && head.y==arrworm[i].y) //если координаты головы червя равны с его хвостом то значит он себя сьел

clearInterval(game,audioGameOver.play(),drawlose()); //проигрыш

}

}

function drawpg(){ //функция отрисовки игры

ctx.drawImage(pg, 0, 0); //отрисовуем фон игры

ctx.drawImage(playground, 0, 0); //отрисовуем игровое поле

ctx.drawImage(foodpic, food.x, food.y); //отображение еды

for(let i = 0; i < worm.length; i++) //червяк

{

ctx.fillStyle = i == 0 ? "pink" : localStorage.getItem('wormcolor'); //если i=0 то голова розовая а иначе все остальные елементы будут цвета выбраного пользователем

ctx.fillRect(worm[i].x, worm[i].y, boxX,boxY); //отображение червяка

}

//счетчик

ctx.fillStyle="white"; //цвет счетчика

ctx.font="75px Arial"; //шрифт и размер

ctx.fillText(score, boxX\*1.5, boxY\*1.90); //координаты размещения

//отрисовка передвежения

let wormX=worm[0].x; //координата первого елемента червя

let wormY=worm[0].y;

if(wormX == food.x && wormY == food.y) { //взаемодействие червя с едой

score++;

audioEat.play()

food = { //радном спавн еды после того как чевяк съел

x:Math.floor((Math.random() \* 17+1))\*boxX,

y:Math.floor((Math.random() \* 15+3))\*boxY,

};

} else

worm.pop();//возврат(удаляем) последнего елемента

if(wormX < boxX || wormX > boxX \* 17 //если вышли за поле проигрыш

|| wormY < 3 \* boxY || wormY > boxY \* 17){

clearInterval(game,audioGameOver.play(),); //очистка после проигриша

drawlose(); //вывод на экран функции с картинкой "GameOver"

}

if(bob == "left") wormX -= boxX; //проверки

if(bob == "right") wormX += boxX;

if(bob == "up") wormY -= boxY;

if(bob == "down") wormY += boxY;

let newHead={ //отрисовка головы

x:wormX,

y:wormY

document.addEventListener('keydown', function(event) { //событие обновления страници при нажатии на пробел

if (event.code == "Space" ) {

location.reload();

}

})

wormtail(newHead,worm);//вызываем функцию

worm.unshift(newHead);//+ элемент

}

let game = setInterval(drawpg,localStorage.getItem('speedworm'));// вызываем функию "drawbg" каждые 100мс для появления в html(скорость перемещения змейки)

playground.setAttribute('src',localStorage.getItem('bgcolor'));//замена цвета игрового поля

foodpic.setAttribute('src',localStorage.getItem('foodcolor'));//замена картинки еды

# ДОДАТОК B