**Cuprins**

Introducere ........................................................................................................... 2

1. Descrierea sarcinii ............................................................................................ 3

2. Algoritmi de funcționare a produsului soft ...................................................... 4

3. Descrierea componentelor utilizate pentru realizarea funcțiilor ...................... 5

4. Descrierea procesului de elaborare a interfeței ................................................ 6

5. Descrierea rezultatului rulării produsului soft ................................................. 7

6. Elaborarea ghidului utilizatorului .................................................................... 8

Concluzii .............................................................................................................. 9

Bibliografie .......................................................................................................... 10

Anexă ................................................................................................................... 11

Mod.

Coala.

№ Document

Semnat

Data

Coala.

2

SDE 525.1 131 02 PA

Executat.

Chirita Mihai

Controlat.

Pocotilenco V.

T.Contr.

Aprobat.

Litera.

Coli

UTM FET E-131

Codul sursă complet al acestui proiect, inclusiv și acest document, sunt disponibile pe **https://github.com/mihai996/unblockme**

**Introducere**

Mulți dintre cei care vor să-și pună mintea la încercare preferă să-și ocupe timpul liber cu dezlegarea unor jocuri, deseori cu un grad de dificultate ridicat. Destul de multe persoane își petrec o parte a timpului liber încercând să rezolve jocuri logice.

Mod.

Coala

№ Document

Semnat

Data

Лист

3

SDE 525.1 131 02 PA

Multe dintre jocurile de logică sunt provocatoare și necesită încerări reptate în scopul de a găsi rezolvarea.

Persistența necesară pentru a rezolva jocurile de inteligență și logică, după eșecuri repetate constituie o lecție de viață pentru copii. Pe parcursul vieții se vor întâlni cu siguranță de situații similare, în care nu o să reușească pentru prima dată și atunci o să știe că trebuie să încerce din nou și astfel vor reuși într-un final.

În această lucrare este propus un joc de logică inspirat din versiunea lui fizică, întâlnit cu numele de „Unblock me”. În acest joc jucătorul trebuie să facă drum liber piesei geometrice de culoare roșie care este blocată în acest labirint din piese de diferite forme geometrice amplasate în așa fel ca să fie posibilă eliberarea piesei roșii din labirint numai după succesivitatea corectă a mișcărilor pieselor, fiind posibilă abaterea de la succesivitatea de bază cu 2-3 mișcări pentru a ajunge la câștig.

**1. Descrierea sarcinii**

Scopul acestei lucrări este de a realiza versiunea digitală a jocului, adică rezultatul să fie cât mai aproape de realitate luînd în considerație limitele tehnice de interacțiune între utilizator și calculator. Pentru a imita procesul de mișcare a pieselor cu un deget, la PC se va utiliza doar mouseul, iar la smartphone touchcreen-ul. Interfața deasemenea va fi cât mai aproape de versiunea fizică, fiecare piesă având culori asemănătoare pieselor reale.

Alegerea tehnologiilor pentru implimentare este o parte nu mai puțin importantă. Am ales tehnologiile web din următoarele motive:

1. Compatibilitatea jocului pe diferite dispozitive (PC, tabletă, smartphone), diferite sisteme de operare (windows, ubuntu, android, iOS), diferite browsere (chrome, firefox, opera).

2. Posibilitatea utilizării aceleiași baze de date.

3. Utilizarea aceleiași interfețe grafice.

4. Nu este necesar de a instala jocul. Este nevoie doar de conexiunea la internet.

Pentru dezvoltarea jocului am utilizat următoarele limbaje: PHP – pentru routing și generarea mapelor; JavaScript – logica jocului, realizarea interacțiunii utilizatorului cu elementele jocului; HTML și CSS – pentru interfața grafică.

În decursul dervoltării proiectului este nevoie de un set de instrumente software precum editor de cod (Sublime Text 2), server local (LAMP), server remote (000webhost.com).

Sublime Text 2 este un editor de cod destinat pentru majoritatea limbajelor de programare, printre care și cele utilizate în proiect. Dispune de multe posibilități precum autoformatarea textului, autocolorarea textului în dependență de sintaxa limbajului, care duc la facilitarea și accelerarea scrierii de cod, folosind template-uri predefinite.

Mod.

Coala

№ Document

Semnat

Data

Лист

4

SDE 525.1 131 02 PA

LAMP (Linux Apache MySQL PHP) este un complex de programe frecvent utilizat pentru a rula aplicații web. Linux – sistemul de operare, Apache – web server, MySQL – server pentru baza de date, PHP – limbaj de programare de script. Analog pentru LAMP putem utiliza WAMP (Windows Apache MySQL PHP) ori XAMP – versiunea cross-platform. Deasemenea pentru o simulare de host remote putem folosi host virtual, care se poate de setat ușor în Apache.

000webhost.com – un service care oferă host, domen, acces prin FTP la host pentru a încărca fișierele necesare, în proiect este folosit pentru testare la distanță de pe diferite dispozitive.

Pentru exemplele din compartimentele următoare vom utiliza adresa „http://myway4u.000webhostapp.com/”.

**2. Algoritmi de funcționare a produsului soft**

Mod.

Coala

№ Document

Semnat

Data

Лист

5

SDE 525.1 131 02 PA

Funcționarea produsului poate fi divizată în diferite compartimente în dependență de procesele care se petrec, de exemplu logica de interacțiune cu user-ul în decursul jocului, logica de acces și schimb de date între browser și server.

Accesarea jocului

1. HTTP GET Request la adresa <http://myway4u.000webhostapp.com/>

2. Browserul primește codul HTML, CSS, JavaScript.

3. Browserul afișează codul interpretat sub formă grafică (text stilizat, contururi, imagini etc.).

4. User-ul îndeplinește o acțiune (click pe link, ori buton), în dependență de ce alege să facă.

4. Ieșirea din joc se efectuează închizând fereastra curentă din browser.

Logica back-end a aplicației

1. Receptarea request-ului.

2. Determinarea dinamică a adresei din request cu ajutorul PHP

3. Pregătirea datelor necesare, încărcarea fișierelor necesare.

4. Răspuns către browser în format HTML, CSS și JavaScript.

Logica jocului

1. Acesarea meniului și alegerea mapei dintre cele accesibile

2. Jocul propriuzis: mutarea blocurilor pînă când piesa roșie ajunge la destinație.

În continuare sunt descrise mai detaliat unele din subpunctele precedente.

- Din logica back-end, determinarea dinamică a adresei cerute are loc în felul următor:

1. Apache caută în mapa solicitată prezența fișierului /.htaccess și urmează regulile înscrise acolo. În cazul nostru dacă adresa corespunde cu numele unui fișier, atunci se îndeplinește script-ul din fișier ori se afișează conținutul lui, în caz contrar celelalte request-uri sunt redirecționate la /index.php, unde scriptul preia responsabilitatea asupra lui.

2. În /index.php se inițiază sesiunea pentru utilizator, se încarcă controlerele necesare, se delimitează cuvintele cheie din URL, se formează rutele și se încarcă fișierele cu sau fără mapele cerute.

- Din logica jocului

1. Userul primește câmpul de joc în versiunea grafică, în care poate interacționa doar cu mouseul ori touchscreen-ul.

Mod.

Coala

№ Document

Semnat

Data

Лист

6

SDE 525.1 131 02 PA

2. La încercarea de a mișca una din piese scriptul determină direcțiile permise de deplasare în funcție de tipul piesei (orizontală ori verticală), poziția ei față de hotarul jocului și poziția ei față de celelalte piese.

3. Dacă legile din punctul precedent convin, atunci la continuarea încercării de a mișca piesa ea se va muta în locul următor conform direcției arătate de user.

4. Verificările din punctul 2 se îndeplinesc în cîteva zeci de milisecunde, așa că utilizatorul nu reușește să observe cum calculatorul ia decizia, iar distanța parcursă cu cursorul de la locul inițial pînă ca piesa să se mute este de cîțiva pixeli.

5. În momentul în care piesa roșie ajunge în locul destinat, este emisă o alertă informativă în care se spune că mapa a fost trecută cu succes, iar după confirmare jocul trece automat la următoarea mapă, făcându-se redirect la URL corespunzător, înregistându-se în sesiune ultima mapă trecută cu succes.

**3. Descrierea componentelor utilizate pentru realizarea funcțiilor**

Fiecare soft sau limbaj utilizat își are destinația proprie în acest produs.

Complexul LAMP este pentru rularea serverului și executarea scriptului PHP. Descrierea detaliată a procesului de instalare și funcționare iese din hotarele lucrării de față.

PHP (PHP: Hypertext Preprocessor) este un limbaj de programare. Folosit inițial pentru a produce pagini web dinamice, este folosit pe scară largă în dezvoltarea paginilor și aplicațiilor web. Se folosește în principal înglobat în codul HTML, dar începând de la versiunea 4.3.0 se poate folosi și în mod „linie de comandă” (CLI), permițând crearea de aplicații independente. Este unul din cele mai importante limbaje de programare web open-source și server-side, existând versiuni disponibile pentru majoritatea web serverelor și pentru toate sistemele de operare. Conform statisticilor este instalat pe 20 de milioane de site-uri web și pe 1 milion de servere web. Este disponibil sub Licenṭa PHP ṣi Free Software Foundation îl consideră a fi un software liber.

HyperText Markup Language (HTML) este un limbaj de marcare utilizat pentru crearea paginilor web ce pot fi afișate într-un browser (sau navigator). Scopul HTML este mai degrabă prezentarea informațiilor – paragrafe, fonturi, tabele ș.a.m.d. – decât descrierea semanticii documentului. Specificațiile HTML sunt dictate de World Wide Web Consortium (W3C).

Mod.

Coala

№ Document

Semnat

Data

Лист

7

SDE 525.1 131 02 PA

CSS (Cascading Style Sheets) este un standard pentru formatarea elementelor unui document HTML. Stilurile se pot atașa elementelor HTML prin intermediul unor fișiere externe sau în cadrul documentului, prin elementul <style> și/sau atributul style. CSS se poate utiliza și pentru formatarea elementelor XHTML, XML și SVGL.

JavaScript (JS) este un limbaj de programare orientat obiect bazat pe conceptul prototipurilor. Este folosit mai ales pentru introducerea unor funcționalități în paginile web, codul Javascript din aceste pagini fiind rulat de către browser. Limbajul este binecunoscut pentru folosirea sa în construirea siturilor web.

jQuery este o platformă de dezvoltare JavaScript, concepută pentru a ușura și îmbunătăți procese precum traversarea arborelui DOM în HTML, managementul inter-browser al evenimentelor, animații și cereri tip AJAX. jQuery a fost gândit să fie cât mai mic posibil, disponibil în toate versiunile de browsere importante existente.

Cum a fost specificat în capitolul precedent, fiecare limbaj și-a primit rolul lui. PHP răspunde de următoarele:

1. Routing (/index.php, /controllers/Router.php)

2. Elaborarea mapelor (/index.php, /maps/\*)

3. Managementul sesiunii

HTML formează baza obiectelor și a pieselor jocului, le grupează și le memorizează codul identificator. CSS (/css/style.css) stilizează toate aceste obiecte, le dă culoare, dimensiune, le poziționează etc.

JavaScript are rolul de a face interacțiunea dintre user și browser conform logicii scrise în acest limbaj. Logica jocului este în fișierul /js/game.js, și poate fi divizată în compartimentele:

1. Inițializarea variabilelor

2. Recepționarea evenimentelor (click pe piese)

3. Prelucrarea evenimentelor în dependență de faza click-ului (down, up, move)

4. Verificarea și calcularea variantelor evenimentelor admisibile

5. Redesenarea mapei conform acțiunilor user-ului

**4. Descrierea procesului de elaborare a interfeței**

Elaborarea interfeței este doar prin scrierea de cod, fără drag and drop ori desenare ori colorare ca în programele de editare a imaginilor. În primul rând se crează baza HTML, se definește diviziunea jocului, butoanele. Apoi cu ajutorul CSS se dă culoare obiectelor, se amplasează în locul potrivit, se colorează, se face contur, se setează imaginea de fundal. În dependență de atributele elementelor din HTML și a regulilor din /css/style.css, browser-ul face concordanța dinamic, adică dacă se schimbă vreun atribut, browser-ul face schimbarea îndată conform regulilor.

Mod.

Coala

№ Document

Semnat

Data

Лист

8

SDE 525.1 131 02 PA

Mărimea și design-ul suprafeței de joc este o imitare a jocului fizic. De exemplu, fiecare piesă din joc este un <div> care primește atributul ’class’ individual, după care CSS stilizează și aranjează obiectul:

*.block.vertical {*

*width: 70px;*

*/\* ... \*/*

*background: linear-gradient(to right, #E0B070, #FFCF8F, #E0B070, #FFCF8F, #E0B070, #FFCF8F, #E0B070); /\* Standard syntax \*/*

*}*

*.block.vertical.v2 {*

*height: 140px;*

*}*

*.block.x2 {*

*left: 140px;*

*}*

De exemplu, *.block.vertical.v2* am utilizat pentru a specifica înălțimea pentru blocurile de lunimea a 2 lățimi, 70px\*2. Class precum *x2* sau *y3* am utilizat pentru a specifica poziția elementului față de marginea de sus și marginea din stânga a câmpului de joc.

**5. Descrierea rezultatului rulării produsului soft**

Mod.

Coala

№ Document

Semnat

Data

Лист

9

SDE 525.1 131 02 PA

Cum a fost menționat în compartimentele precedente, pentru a putea rula jocul este nevoie ca codul sursă să ruleze pe un server compatibil cu tehnologiile de back-end utilizate, de un browser compatibil cu tehnologiile front-end utilizate.

La accessarea link-ului respectiv utilizatorului îi apare deodată meniul principal cu 2 variante de alegere: „New game” – joc nou, și „Maps” – alegerea mapei dorite dacă numărul ei este mai mic sau egal cu numărul mapei deschise cu numărul de ordine cel mai mare. Fiecare următoare mapă se deschide doar la rezolvarea mapei precedente. Mapa 1 este disponibilă din start.

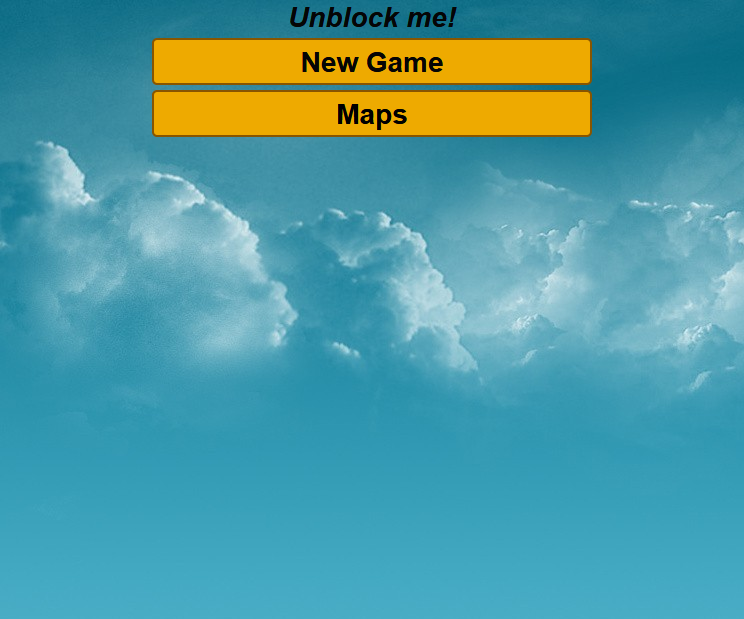


Fig. 1. Prima rulare a jocului

După selectarea mapei utilizatorul are în față câmpul de joc unde trebuie să mute blocurile în așa fel ca cel roșu să ajungă în locul necesar.

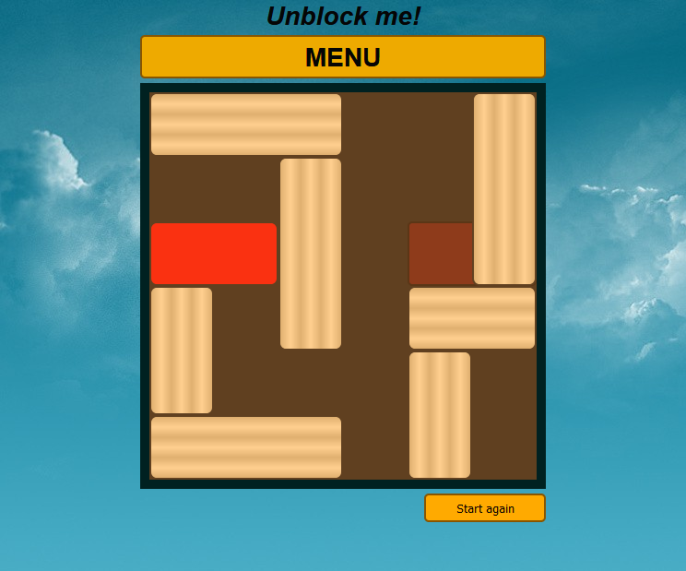


Fig. 2. Mapa 1

**6. Elaborarea ghidului utilizatorului**

Creatorii și dezvoltatorii de jocurilor au ca țintă utilizatori dintr-un diapazon foarte mare, începând de la cei cu cunoștințe minime în utilizarea calculatorului, de aceea accesarea jocului și interacțiunea cu el trebuie să fie cât mai simplă și intuitivă posibilă. Reieșind din aceasta accesarea are loc prin url, ca orice site, iar în timpul jocului de obicei este limitat la 2-3 butoane pe care este scris destinația lui, precum „New game”, „Menu”, „Start again”, iar însuși procesul jocului este imitarea jocului real, ceea ce nu ar trebui să fie o piedică pentru nici un om cu vârsta peste 5 ani, partea dificilă fiind doar căutarea mișcărilor potrivite.

Mod.

Coala

№ Document

Semnat

Data

Лист

10

SDE 525.1 131 02 PA

**Concluzii**

Mod.

Coala

№ Document

Semnat

Data

Лист

11

SDE 525.1 131 02 PA

Elaborând această lucrare de an am aflat lucruri noi și mi-am întărit cunoștințele în dezvoltarea de aplicații software. Am utilizat diferite limbaje și tehnologii pentru a duce la bun sfârșit scopul propus. Am făcut funcții de automatizare a lucrului proiectului, precum routingul, încărcarea dinamică a fișierelor de tip *view*, încărcarea dinamică a mapelor. Am implimentat posibilitatea de a juca cu mouse și cu touch-screen.

Ce ține de jocul propriuzis, am încercat să imit mișcarea lină a blocurilor însă din cauza că în unele cazuri (de obicei calculator lent sau supraîncărcat) JavaScript poate să ruleze cu o viteză mai mică decât utilizatorul poate să întreprindă care acțiuni, precum mișcarea bruscă a mouselui cu suprapunere de clickuri, ceea ce duce la producerea unor erori în regulile jocului, precum ieșirea blocului din câmpul de joc sau suprapunerea cu alte blocuri. De aceea, propun, pentru o performanță mai bună a jocului pentru aceste cazuri, de a recurge la alte tehnologii ori de a mai limita puțin posibilitățile de acțiune a utilizatorului.

Pe parcursul dezvoltării proiectului am creat de funcții de debug, pentru a vedea rezultate intermediare, pentru a urmări starea câmpului de joc și pentru a vedea erorile intervenite în cod. Am realizat funcții de debug în php și în javascript, iar pentru versiunea live, am inclus ieșire imediată din aceste funcții pentru o performanță mai bună și pentru a nu arunca tot „gunoiul” spre ochii utilizatorului.

**Bibliografie**

Mod.

Coala

№ Document

Semnat

Data

Лист

12

SDE 525.1 131 02 PA

1. <http://php.net/manual/ro/>

2. <https://www.w3schools.com/>

3. <https://developer.mozilla.org/bm/docs/Web/JavaScript>

4 . <https://ro.wikipedia.org/wiki/LAMP>

5. <https://ro.wikipedia.org/wiki/PHP>

6. <https://ro.wikipedia.org/wiki/HyperText_Markup_Language>

7. <https://ro.wikipedia.org/wiki/Cascading_Style_Sheets>

**Anexă**

Mod.

Coala

№ Document

Semnat

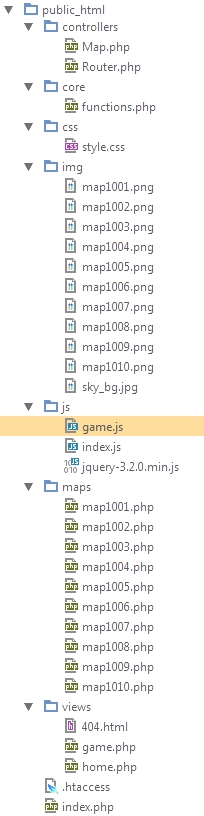
Data

Лист

12

SDE 525.1 131 02 PA

Anexa 1. Structura proiectului



Anexa 2. Controllers/Map.php

<?php

function getMap($n = 1)

Mod.

Coala

№ Document

Semnat

Data

Лист

13

SDE 525.1 131 02 PA

{

$file = APP\_PATH . "/maps/map" . (1000 + $n) . ".php";

if (file\_exists($file)) {

include $file;

} else {

$file = APP\_PATH . "/maps/map1001.php";

include $file;

}

return isset($m) ? json\_encode($m) : '0';

}

function maps()

{

$count = 1;

$check\_next = true;

while ($check\_next) {

if (file\_exists(APP\_PATH . "/maps/map" . (1000 + $count) . ".php")) {

$count++;

} else {

$check\_next = false;

}

}

return $count;

}

?>

Anexa 3. Controllers/Router.php

Mod.

Coala

№ Document

Semnat

Data

Лист

14

SDE 525.1 131 02 PA

<?php

class Router

{

public static function getCurrentUri()

{

$basepath = implode('/', array\_slice(explode('/', $\_SERVER['SCRIPT\_NAME']), 0, -1)) . '/';

$uri = substr($\_SERVER['REQUEST\_URI'], strlen($basepath));

if (strstr($uri, '?')) {

$uri = substr($uri, 0, strpos($uri, '?'));

}

$uri = '/' . trim($uri, '/');

return $uri;

}

public static function getRoutesArray()

{

$base\_url = self::getCurrentUri();

$routes = array();

$url\_exploded = explode('/', $base\_url);

foreach ($url\_exploded as $route) {

if (trim($route) != '') {

array\_push($routes, ($route));

}

}

return $routes;

}

}

?>

Anexa 4. core/functions.php

Mod.

Coala

№ Document

Semnat

Data

Лист

15

SDE 525.1 131 02 PA

<?php

function qwe($r)

{

echo "<pre>";

var\_dump($r);

echo "</pre>";

}

function loadView($view, $data)

{

if (@count($data) > 0) {

extract($data, EXTR\_OVERWRITE);

}

ob\_start();

include\_once(APP\_PATH . '/views/' . $view);

$content = ob\_get\_contents();

ob\_end\_clean();

echo $content;

}

?>

Anexa 5. maps/map1001.php

Mod.

Coala

№ Document

Semnat

Data

Лист

16

SDE 525.1 131 02 PA

<?php

// MAP 01

$m = array(

array('id' => 'block00', 'type' => 'horizontal', 'type\_short' => 'h', 'len' => 2, 'x' => 0, 'y' => 2)

, array('id' => 'block01', 'type' => 'horizontal', 'type\_short' => 'h', 'len' => 3, 'x' => 0, 'y' => 0)

, array('id' => 'block02', 'type' => 'horizontal', 'type\_short' => 'h', 'len' => 2, 'x' => 4, 'y' => 3)

, array('id' => 'block03', 'type' => 'horizontal', 'type\_short' => 'h', 'len' => 3, 'x' => 0, 'y' => 5)

, array('id' => 'block04', 'type' => 'vertical', 'type\_short' => 'v', 'len' => 2, 'x' => 0, 'y' => 3)

, array('id' => 'block05', 'type' => 'vertical', 'type\_short' => 'v', 'len' => 3, 'x' => 2, 'y' => 1)

, array('id' => 'block06', 'type' => 'vertical', 'type\_short' => 'v', 'len' => 2, 'x' => 4, 'y' => 4)

, array('id' => 'block07', 'type' => 'vertical', 'type\_short' => 'v', 'len' => 3, 'x' => 5, 'y' => 0)

);

?>

Anexa 6. views/game.php

<?php

if(!empty($\_GET['game']) && $\_GET['game'] == 'new'){

session\_destroy();

Mod.

Coala

№ Document

Semnat

Data

Лист

17

SDE 525.1 131 02 PA

session\_start();

}

?>

<!DOCTYPE html>

<html>

<head>

<title>Map <?php echo $map ?> | Unblock me!</title>

<meta name="viewport" content="width=500, initial-scale=0.7, maximum-scale=0.7">

<link rel="stylesheet" type="text/css" href="/css/style.css">

<script src="/js/jquery-3.2.0.min.js"></script>

</head>

<body>

<div class="body">

<div class="main">

<input type="hidden" id="map\_id" value='<?php echo $map; ?>'>

<input type="hidden" id="blocks" value='<?php echo getMap($map); ?>'>

<a href="/"><div class="title"><i>Unblock me!</i></div></a>

<div class="menu">

<a href="/"><div class="item">MENU</div></a>

</div>

<div class="game">

<div class="destination"></div>

</div>

<div class="buttons">

<button type='button' id="refreshGame">Start again</button>

</div>

<div><pre id="debug"></pre></div>

</div>

</div>

<script src="/js/game.js"></script>

</body>

</html>

Anexa 7. views/home.php

Mod.

Coala

№ Document

Semnat

Data

Лист

18

SDE 525.1 131 02 PA

<!DOCTYPE html>

<html>

<head>

<title>Unblock me!</title>

<meta name="viewport" content="width=500, initial-scale=0.7, maximum-scale=0.7">

<link rel="stylesheet" type="text/css" href="/css/style.css">

<script src="/js/jquery-3.2.0.min.js"></script>

<script src="/js/index.js"></script>

</head>

<body>

<div class="body">

<div class="main">

<a href="/"><div class="title"><i>Unblock me!</i></div></a>

<div class="text error"><?php echo !empty($error) ? $error : ''; ?></div>

<div class="menu">

<?php $maps\_count = maps(); ?>

<?php if (!empty($\_SESSION['last\_game']) && $\_SESSION['last\_game'] < $maps\_count && $\_SESSION['last\_game'] > 1) { ?>

<a href="/game/<?php echo $\_SESSION['last\_game'] ?>">

<div class="item">Continue</div>

</a>

<?php } ?>

<a href="/game/1?game=new"><div class="item">New Game</div></a>

<div id="all\_maps\_btn" class="item">Maps</div>

</div>

<div class="all\_maps" style="display: none;">

<ul>

<?php for ($i = 1; $i < $maps\_count; $i++) {

if (!empty($\_SESSION['last\_game']) && $\_SESSION['last\_game'] >= $i) {

echo '<a href="/game/' . $i . '"><li style=" background-image: url(/img/map' . (1000 + $i) . '.png)"></li></a>';

} else {

echo '<li style="opacity: 0.5; background-image: url(/img/map' . (1000 + $i) . '.png)"></li>';

}

} ?>

</ul>

</div>

</div>

</div>

</body>

</html>

Anexa 8. index.php

<?php

session\_start();

if (empty($\_SESSION['last\_game'])) {

Mod.

Coala

№ Document

Semnat

Data

Лист

19

SDE 525.1 131 02 PA

$\_SESSION['last\_game'] = 1;

}

require\_once 'controllers/Router.php';

require\_once 'controllers/Map.php';

require\_once 'core/functions.php';

define('APP\_PATH', getcwd());

$route = Router::getRoutesArray();

if (@count($route) == 0) {

loadView('home.php', array());

} else if (@count($route) == 1) {

if ($route[0] == 'save\_game' && !empty($\_POST['save\_game'])) {

$\_SESSION['last\_game'] = $\_POST['save\_game'];

} else {

loadView('home.php', array());

}

} else if (@count($route) == 2) {

if ($route[0] == 'game') {

if ((int)$route[1] > 0 && (int)$route[1] < maps()) {

loadView('game.php', array('map' => (int)$route[1]));

} else {

loadView('home.php', array('error' => 'Map ' . (int)$route[1] . ' is not created yet...'));

}

} else {

loadView('home.php', array('error' => ''));

}

} else {

loadView('home.php', array());

}

?>

Anexa 9. js/game.js

Mod.

Coala

№ Document

Semnat

Data

Лист

20

SDE 525.1 131 02 PA

$(document).ready(function () {

var pixel\_diff = 30;

var isDown = false;

var mouseMovePosition = {x: '', y: ''};

var mouseDownPosition = {x: '', y: ''};

var diff\_position = {x: '', y: ''};

var rel\_position\_init = {x: '', y: ''};

var rel\_position\_curr = {x: '', y: ''};

var clickedBlock = {id: '', elem: ''};

var block\_id\_int = '';

var block\_type = '';

var map = [

[0, 0, 0, 0, 0, 0],

[0, 0, 0, 0, 0, 0],

[0, 0, 0, 0, 0, 0],

[0, 0, 0, 0, 0, 0],

[0, 0, 0, 0, 0, 0],

[0, 0, 0, 0, 0, 0]

];

var blocks;

var response;

var init\_map;

var init\_blocks;

response = $('#blocks').val();

var map\_id = parseInt($('#map\_id').val());

blocks = JSON.parse(response);

init\_map = JSON.parse(JSON.stringify(map));

init\_blocks = JSON.parse(JSON.stringify(blocks));

drawMap();

function eMouseDown(e) {

var id\_patt = /^block../;

if (id\_patt.test($(this).attr('id'))) {

isDown = true;

clickedBlock.elem = $(this);

clickedBlock.id = clickedBlock.elem.attr('id');

if (e.type == 'touchstart') {

mouseDownPosition.x = e.touches[0].clientX;

mouseDownPosition.y = e.touches[0].clientY;

mouseMovePosition.x = e.touches[0].clientX;

mouseMovePosition.y = e.touches[0].clientY;

} else {

mouseDownPosition.x = e.clientX;

mouseDownPosition.y = e.clientY;

mouseMovePosition.x = e.clientX;

mouseMovePosition.y = e.clientY;

}

rel\_position\_init.x = clickedBlock.elem.position().left;

Mod.

Coala

№ Document

Semnat

Data

Лист

21

SDE 525.1 131 02 PA

rel\_position\_init.y = clickedBlock.elem.position().top;

block\_id\_int = parseInt(clickedBlock.id.substr(5, 2));

block\_type = blocks[parseInt(clickedBlock.id.substr(5, 2))].type;

rel\_position\_curr.x = clickedBlock.elem.position().left;

rel\_position\_curr.y = clickedBlock.elem.position().top;

} else {

isDown = false;

}

}

function eMouseUp() {

emptyVars();

isDown = false;

if (blocks[0].x == 4) {

$.ajax({

url: '/save\_game',

type: 'POST',

data: {save\_game: 1 + map\_id}

});

alert('Congratulation!');

window.location = '/game/' + (1 + map\_id);

}

}

function eMouseMove(e) {

//e.stopPropagation();

//e.preventDefault();

debug();

if (isDown) {

if (e.type == 'touchmove') {

mouseMovePosition.x = e.touches[0].clientX;

mouseMovePosition.y = e.touches[0].clientY;

} else {

mouseMovePosition.x = e.clientX;

mouseMovePosition.y = e.clientY;

}

rel\_position\_curr.x = clickedBlock.elem.position().left;

rel\_position\_curr.y = clickedBlock.elem.position().top;

diff\_position.x = mouseMovePosition.x - mouseDownPosition.x;

diff\_position.y = mouseMovePosition.y - mouseDownPosition.y;

if (block\_type == 'horizontal') {

moveHBlock();

} else if (block\_type == 'vertical') {

Mod.

Coala

№ Document

Semnat

Data

Лист

22

SDE 525.1 131 02 PA

moveVBlock();

}

}

}

$(document).on('mousedown', '.block', eMouseDown);

$(document).on('touchstart', '.block', eMouseDown);

document.addEventListener('mouseup', eMouseUp);

document.addEventListener('touchend', eMouseUp);

document.addEventListener('mousemove', eMouseMove);

document.addEventListener('touchmove', eMouseMove);

function checkMap(type) {

debug();

if (type == 'H') {

if (diff\_position.x > 0) { // to right

var next\_cell = blocks[block\_id\_int].x + blocks[block\_id\_int].len;

if (next\_cell <= 5

&& map[blocks[block\_id\_int].y][next\_cell] == 0

) {

return true;

} else {

return false;

}

} else if (diff\_position.x < 0) { // to left

var next\_cell = blocks[block\_id\_int].x - 1;

if (next\_cell >= 0

&& map[blocks[block\_id\_int].y][next\_cell] == 0

) {

return true;

} else {

return false;

}

}

} else if (type == 'V') {

if (diff\_position.y > 0) { // down

var next\_cell = blocks[block\_id\_int].y + blocks[block\_id\_int].len;

if (next\_cell <= 5

&& map[next\_cell][blocks[block\_id\_int].x] == 0

) {

return true;

} else {

return false;

}

} else if (diff\_position.y < 0) { // up

var next\_cell = blocks[block\_id\_int].y - 1;

if (next\_cell >= 0

&& map[next\_cell][blocks[block\_id\_int].x] == 0

) {

return true;

} else {

Mod.

Coala

№ Document

Semnat

Data

Лист

23

SDE 525.1 131 02 PA

return false;

}

}

}

return false;

}

function moveHBlock() {

debug();

if (checkMap('H')) {

var refresh = false;

if (diff\_position.x >= pixel\_diff) { // to right

map[blocks[block\_id\_int].y][blocks[block\_id\_int].x] = 0;

map[blocks[block\_id\_int].y][blocks[block\_id\_int].x + blocks[block\_id\_int].len] = 1;

blocks[block\_id\_int].x += 1;

refresh = true;

} else if (diff\_position.x <= -1 \* pixel\_diff) { // to left

map[blocks[block\_id\_int].y][blocks[block\_id\_int].x + blocks[block\_id\_int].len - 1] = 0;

map[blocks[block\_id\_int].y][blocks[block\_id\_int].x - 1] = 1;

blocks[block\_id\_int].x -= 1;

refresh = true;

}

if (refresh) {

mouseDownPosition.x = mouseMovePosition.x;

mouseDownPosition.y = mouseMovePosition.y;

refreshBlock();

}

//clickedBlock.elem.css('left', rel\_position\_init.x + diff\_position.x);

//clickedBlock.elem.css('left', rel\_position\_init.x + Math.floor(diff\_position.x/10.0)\*10 + 'px');

}

}

function moveVBlock() {

debug();

if (checkMap('V')) {

var refresh = false;

if (diff\_position.y >= pixel\_diff) { // down

map[blocks[block\_id\_int].y][blocks[block\_id\_int].x] = 0;

map[blocks[block\_id\_int].y + blocks[block\_id\_int].len][blocks[block\_id\_int].x] = 1;

Mod.

Coala

№ Document

Semnat

Data

Лист

24

SDE 525.1 131 02 PA

blocks[block\_id\_int].y += 1;

refresh = true;

} else if (diff\_position.y <= -1 \* pixel\_diff) { // up

map[blocks[block\_id\_int].y + blocks[block\_id\_int].len - 1][blocks[block\_id\_int].x] = 0;

map[blocks[block\_id\_int].y - 1][blocks[block\_id\_int].x] = 1;

blocks[block\_id\_int].y -= 1;

refresh = true;

}

if (refresh) {

mouseDownPosition.x = mouseMovePosition.x;

mouseDownPosition.y = mouseMovePosition.y;

refreshBlock();

}

//clickedBlock.elem.css('top', rel\_position\_init.y + diff\_position.y);

//clickedBlock.elem.css('top', rel\_position\_init.y + (diff\_position.y > 0 ? '70' : '-70') + 'px');

}

}

function refreshBlock() {

debug();

if (blocks[block\_id\_int].type\_short == 'h') {

$('#' + clickedBlock.id).attr('class', 'block ' + blocks[block\_id\_int].type + ' ' + blocks[block\_id\_int].type\_short + blocks[block\_id\_int].len + ' x' + blocks[block\_id\_int].x + ' y' + blocks[block\_id\_int].y);

} else if (blocks[block\_id\_int].type\_short == 'v') {

$('#' + clickedBlock.id).attr('class', 'block ' + blocks[block\_id\_int].type + ' ' + blocks[block\_id\_int].type\_short + blocks[block\_id\_int].len + ' x' + blocks[block\_id\_int].x + ' y' + blocks[block\_id\_int].y);

}

}

function drawMap() {

Mod.

Coala

№ Document

Semnat

Data

Лист

25

SDE 525.1 131 02 PA

$('.game').html('<div class="destination"></div>');

for (var i = 0; i < blocks.length; i++) {

var blockClass = ' ' + blocks[i].type + ' ' + blocks[i].type\_short + blocks[i].len + ' x' + blocks[i].x + ' y' + blocks[i].y;

$('.game').append('<div id="' + blocks[i].id + '" class="block ' + blockClass + ' "> </div>');

if (blocks[i].type\_short == 'h') {

for (var j = blocks[i].x; j < blocks[i].x + blocks[i].len; j++) {

map[blocks[i].y][j] = 1;

}

} else {

for (var j = blocks[i].y; j < blocks[i].y + blocks[i].len; j++) {

map[j][blocks[i].x] = 1;

}

}

}

debug();

}

function refreshGame() {

map = [];

map = JSON.parse(JSON.stringify(init\_map));

blocks = [];

blocks = JSON.parse(JSON.stringify(init\_blocks));

emptyVars();

$('.game').html('<div class="destination"></div>');

debug();

drawMap();

}

$(document).on('click', '#refreshGame', function () {

refreshGame();

});

function emptyVars() {

isDown = false;

mouseMovePosition = {x: '', y: ''};

mouseDownPosition = {x: '', y: ''};

diff\_position = {x: '', y: ''};

rel\_position\_init = {x: '', y: ''};

rel\_position\_curr = {x: '', y: ''};

clickedBlock = {id: '', elem: ''};

block\_id\_int = '';

block\_type = '';

}

function debug() {

Mod.

Coala

№ Document

Semnat

Data

Лист

26

SDE 525.1 131 02 PA

// --for debug:

return;

var a = '';

for (var i = 0; i < map.length; i++) {

a += map[i] + "\r\n";

}

a += "\r\n\r\n";

for (var i = 0; i < blocks.length; i++) {

a += '' + 'id: ' + blocks[i].id + ' type\_short: ' + blocks[i].type\_short + ' len: ' + blocks[i].len + ' x: ' + blocks[i].x + ' y: ' + blocks[i].y + "\r\n";

}

a += "\r\n\r\nisDown: " + isDown;

a += "\r\n\r\nblock: " + block\_id\_int;

$('#debug').text(a);

// --end

}

});