HTML5 game
inside Facebook

Honour Project
2012-2013

Roberto Martínez Querol

51234596

Department of Computing Science
University of Aberdeen
Kings’s College
Aberdeen AB24 3UE
Abstract
This project involves the research and creation of a HTML5 Facebook game. A Facebook game is a software application inside the social network Facebook, that means you only can play this games if your logged in Facebook. In this report I will explain what is required to create a game on Facebook, to be more precise I explain how I take two HTML5 games and I put them on Facebook. This games are a single-player game (Snake) and two-players games (Connect4).

Acknowledgement
I would like to thanks my project supervisor, Dr. Bruce Scharlau for his assistance and advice during the development of this project, and would also like to thank my family and friends for supporting me during the development of this project and for helping in the testing phase.

Declaration
I declare that this document and the accompanying code has been composed by myself, and describes my own work, unless otherwise acknowledged in the text. It has not been accepted in any previous application for a degree. All verbatim extracts have been distinguished by quotation marks, and all sources of information have been specifically acknowledged.

Signed:.................................................................

Roberto Martinez Querol        Date: ..........................

Page: 2
# Table of Contents

Abstract.............................................................................................................................................. 2
Acknowledgement..................................................................................................................................... 2
Declaration............................................................................................................................................... 2
1. Introduction........................................................................................................................................ 5
   1.1. Overview....................................................................................................................................... 5
   1.2. Primary goals................................................................................................................................. 5
   1.3. Secondary goals............................................................................................................................. 5
   1.4. Steps............................................................................................................................................... 6
2. Background.......................................................................................................................................... 7
   2.1. Games in Social networks............................................................................................................ 7
   2.2. HTML5 games............................................................................................................................... 7
3. Methodology....................................................................................................................................... 8
4. What has been done until now on Facebook .................................................................................... 9
5. Technologies..................................................................................................................................... 10
   5.1. Hosting......................................................................................................................................... 10
   5.1.1. x10hosting............................................................................................................................... 10
   5.1.2. Heroku...................................................................................................................................... 10
   5.1.3. Others...................................................................................................................................... 10
   5.2. Javascript..................................................................................................................................... 11
   5.2.1. Facebook SDK for javascript................................................................................................. 11
   5.2.2. jQuery...................................................................................................................................... 11
   5.3. HTML5........................................................................................................................................ 13
   5.3.1. Alternative: Adobe Flash Player.............................................................................................. 13
   5.4. CSS............................................................................................................................................. 13
   5.5. Data storage................................................................................................................................. 14
   5.5.1. ClearBD mySQL....................................................................................................................... 14
   5.5.2. Redis cloud (nosql)................................................................................................................ 14
   5.5.3. Others...................................................................................................................................... 15
   5.6. PHP............................................................................................................................................. 16
   5.6.1. Facebook SDK for PHP.......................................................................................................... 16
   5.6.2. predis..................................................................................................................................... 16
   5.6.3. Alternative to PHP.................................................................................................................. 16
   5.6.4. Sequel pro............................................................................................................................... 17
6. Development and Implementation....................................................................................................... 18
   6.1. Start with the app on Facebook.................................................................................................... 18
   6.1.1. Create app .............................................................................................................................. 18
   6.1.2. Submit app.............................................................................................................................. 19
   6.1.3. Heroku.................................................................................................................................... 21
   6.2. Snake.......................................................................................................................................... 23
   6.2.1. What is Snake?......................................................................................................................... 23
   6.2.2. index.php............................................................................................................................... 24
   6.2.3. game.js.................................................................................................................................. 24
   6.2.4. Data base............................................................................................................................... 26
6.3. Connect 4....................................................................................................................................... 27
   6.3.1 What is connect4?..................................................................................................................... 27
1. Introduction

1.1. Overview

In this day many people have Facebook and many of them play Facebook games. The point is the development of these games, each of them is composed of several parts, such as the server, database, etc. and is not so easy to communicate the different parts and neither is easy do them.

This project wants to explain the bases to create each of the parts and the connections between them in order to explain how create a simple game where two people with a Facebook account can play the game against each other.

1.2. Primary goals

The primary goals of the project are as follows:

- Know how work a HTML5 game for Facebook.
- Manage the server to send, receive and store information.
- Manage the communication with Facebook to play with friends and store the games.
- Manage the database to save all the data (sql and nosql).

1.3. Secondary goals

The secondary goals of the project are as follows:

- Make the connections in a complex multiplayer game like a MMORPG (massive multiplayer role playing game) or something like that but it can be hard.
1.4. Steps

The steps to follow are the next:

• Single-player Facebook game, this game store the maximum score.

• Simple two-player Facebook game, storing the state of the game, the times each player win a game and the data necessary for the turns management.

• Multiplayer game for Facebook. (optional)
2. Background

2.1. Games in Social networks
Social networks have become important in our days, and along with them the games offered integrated in them. Facebook is one of the most important social networks and have a lot of games.

Social games now are a real alternative of electronic entertainment. More and more companies knowing the opportunities offered by this new variant of games and use their best efforts to the creation of social networking entertainment. It is logical if we consider the existing demand within these pages, the amount of user they have, the time spent within them and is not a big problem integrated in Facebook.

In August 2012 over 235 million people play games on Facebook, ¼ of Facebook users.[3]

2.2. HTML5 games
HTML5 (HyperText Markup Language) is a markup language for structuring and presenting content for the World Wide Web and a core technology of the Internet. It is one way to develop games for Facebook. At the beginning Flash was used for the development of games for Facebook but now HTML5 can be a better choice.

More Facebook games are in 2D. Now the power for 2D animation is the same thing with HTML5 and Flash, the advantage HTML5 can run in more mobile devices, Flash will not have upgrades, and a lot of big companies are in HTML5 side and are leaving out Flash including Google, Chrome, Firefox, safari and Adobe. The disadvantage is that HTML5 does not work in older browsers.
3. Methodology

At the beginning it is essential to search information about video games in html5 for Facebook so we can investigate the possible technologies that can be used for development of the project. When we have the information is moment to choose the most appropriate technologies in each case.

Once this is done we need to find a hosting to store our game made in HTML5 and when the simple game is working we can go to the next step.

The next step will be to put this game on Facebook using the information provide by Facebook about the person who wants to play, we will store the score of each player. This first game will be one simple, we only store the maximum user score.

After we finish this we should search for a simple multiplayer game, in our case a typical board game where a person play against other person in turns. Will have to adapt the game to keep the information in a database and communicate to each player the game state at each instant of time. For this we must study the best way to do this, once this is done we will put the game in Facebook. In Facebook two friends will can play with each other.

The next logical step is put a MMOG in Facebook. This could become more complicated because this kind of game have more complexity, for this reason this part is optional.

The final part is finish the documentation of the project.
4. What has been done until now on Facebook

Until now most of browser games have been done in Flash. 70% of web-based games are build using Flash, including 24 of the top 25 Facebook games.\[4\]

But now some people are doing the games in HTML5 for Facebook. The reason is maybe Flash announced they will no longer adapt Flash Player for mobile devices to new browser, OS version or device configurations and this is a problem for the developers because the mobile devices are very important in these days and if they don't adapt Flash for the new browser Flash will die early.\[5\]

Also Flash doesn't work in Iphone/Ipad.

For this reason I will implement the game in HTML5, it works in mobile and a lot of browsers and it will continue to be updated then I think is a good idea to work on Facebook.
5. Technologies

5.1. Hosting
The hosting is required for host our game. This host must have a good data-storage and a SSL security because Facebook needs it.

5.1.1. x10hosting
X10hosting is a free hosting site. They have been hosting the masses since 2004. It have a free hosting includes PHP5, MySQL, auto install script such as Joomla, wordpress, phpBB and SMF.[1]
This one is used in the first part of the project, do a simple online game but it is impossible to use it later because this hosting does not have SSL on the free version and Facebook needs it.

5.1.2. Heroku
Heroku is a free hosting site in cooperation with Facebook. Heroku (pronounced her-OH-koo) is a cloud application platform, a new way of building and deploying web apps. Their service lets app developers spend 100% of their time on their application code, not managing servers, deployment, ongoing operations, or scaling. Heroku provides a platform for building, deploying and running cloud apps in many programming languages. They have a lot of add-ons system for extending the capabilities of their platform. [2]
This one is the most important hosting in the project because it provides a SSL service for free and have a lot capabilities and making easier the game creation.

5.1.3. Others
There are many others hosting sites in internet to host web pages, games, information, etc. I will explain some of them below:

- 000webhost.com: Free hosting with uptime guarantee. He has PHP with MySQL Database support. The free version without SSL.[6]
- freehosting.com: Free hosting site, they provide a lot of serviced like: cPanel, E-Mail and WebMail hosting with IMAP, POP3 and SMTP, FTP, CGI, PHP5, MySQL, PYTHON, ROR, CRON and File Manage. Don't have SSL.[12]
5.2. Javascript

Javascript is a programming language that runs on a web browser. It was first implemented in version 2 of the browser. Javascript is not a subset of Java, in fact, the two languages share little in common. It runs on the browser; it's a client-side scripting language.\[7\]

The HTML5 canvas is a region code inside HTML code with height and width. With javascript you can draw on the canvas and this is the way to do a game with HTML5. First we define inside a HTML one canvas element and then we can use the javascript code for modified the canvas element and draw something inside. With this we can do a game inside a HTML webpage. Javascript is the standard for do that.

5.2.1. Facebook SDK for javascript

The Facebook SDK for JavaScript provides a client-side functionality for the management of the applications on Facebook. This functionalities are:

- Enables you to use the Like Button and other Social Plugins on your site.
- Enables you to use Facebook Login to lower the barrier for people to sign up on your site.
- Makes it easy to call into Facebook's primary API, called the Graph API.
- Launch Dialogs that let people perform various actions like sharing stories.
- Facilitates communication when you're building a game or an app tab on Facebook.\[11\]

5.2.2. jQuery

jQuery is a fast, small, and feature-rich JavaScript library. It makes things like HTML document traversal and manipulation, event handling, animation, and Ajax much simpler with an easy-to-use API that works across a multitude of browsers. With a combination of versatility and extensibility, jQuery has changed the way that millions of people write JavaScript.\[8\]

jQuery is used in the application to call PHP pages from JavaScript code. These calls are for the communication with the database to set, get, delete or add information in our data base.
To make the call we use this code structure:

```javascript
$.ajax({ url:'(url where I have the php)',
    data: {name of the variable):(variable content)},
    type: 'post',
    success: function(output) {
        (code to do after the call)
    }
});
```

I always do the post method and the variable “output” has the value of the response returned by the PHP.

With this the php can do the queries as normal.

```php
mysql_query(“(query)”);
```

To know more about this exists more information here: [http://php.net/](http://php.net/).
5.3. HTML5

HTML5 is the new standard HTML (HyperText Markup Language) language for structuring and presenting content on the web. The HTML5 intention is to include all web technologies in a single standard.

\[
\text{HTML + CSS + JavaScript APIs} = \text{HTML5}
\]

The differences between the old HTML and HTML5 are:

- New elements for a specific content: header, nav, footer, article...
- New functionalities through a standardized interface.
- Audio and video elements for a multimedia reproduction.
- New element called canvas to draw on it.
- New inputs like calendar, date, email...

- Born a lot of APIs based in JavaScript: canvas, webstorage, web sockets, XMLHttpRequest2.... \[9\]

5.3.1. Alternative: Adobe Flash Player

The alternative for do games on Facebook is Adobe Flash Player. Flash is a multimedia and software platform used for authoring of vector graphics, animation, games and Rich internet applications which can be viewed, played and executed in Adobe Flash Player.

The problems is they announced they will no longer adapt Flash Player for mobile devices to new browser, OS version or device configurations.\[10\]

Taking this into account, the best option in the development of video games for facebook I think is HTML5.

5.4. CSS

CSS (Cascading Style Sheet) is a style sheet language used for describing the presentation semantics of a document written in a markup languages, in our case HTML5. Is used in our game for do the main page of the game which contains the game information, user information, etc.
5.5. **Data storage**

For the game we need a good data storage with a good management. If the game have a lot people playing if it is not good the database can have problems. Exists a lot of databases for do that, the most common are SQL database but exists others. In Heroku you can find add-ons for data storage for the applications you are storing there.

5.5.1. **ClearBD mySQL**

ClearDB creates multi-master and multi-master with multi- replica MySQL configurations in regions that are important to you to provide your applications with a fully redundant solution that can survive outages, network failures and even natural disasters. You choose how much computing power and storage you need, then they set it all up for you so that you have a secure, powerful and reliable MySQL database that best serves your needs.

Under the hood, ClearDB uses a combination of advanced replication techniques, advanced cluster technology, and layered web services to provide you with a MySQL database that is "smarter" than usual. We also use things like mixed binary replication logging and auto-increment offset seeding so that you can continue using MySQL’s nondeterministic and time-based functions such as UUID(), NOW() as well as auto-increment keys in your tables.[13]

I choose this data base because is a SQL database and I am familiar with this kind of databases and SQL is a very common good way for store and access the data. In Heroku you can find SQL add-ons for data storage (like Heroku postgres) but ClearBD is a good choice for his simplicity and utility. The connections with ClearDB in our applications I made with PHP code.

5.5.2. **Redis cloud (nosql)**

Redis is an open source, advanced key-value store. It works out-of-the-box with Express and many other Node.js apps and modules. The database instances are on a fast local network connection and offer superior speed compared to other hosted solutions.[14]

This solution is faster than the SQL solution, this is visible in the tests.

It has a PHP library, we use this library for do the connections with our redis, the reason is for easy comparison with the SQL option (ClearBD).
5.5.3. Others

Heroku has a lot of add-ons for data storage like:

- **Mongo lab**: Mongo (from the word "humongous") is an open-source database NoSQL. MongoDB stores structured data as JSON-like documents with dynamic schemas (MongoDB calls the format BSON). Mongo lab is a fully-managed cloud database service featuring highly-available MongoDB database, automated backups, web-based tools, 24/7 monitoring, and expert support.[16]

- **MongoHQ**: Is another MongoDB. They provide more or less the same than Mongo lab.

- **RedisGreen**: Is like Redis cloud with different management.

- **Heroku Postgres**: Is a SQL database like ClearBD. PostgreSQL is a data-management system. Open-source object-relational database. Works with: Java, Perl, Python, Ruby, Tcl, C/C++ and its own PL/pgSQL.
5.6. PHP

PHP (Hypertext Preprocessor) is a widely-used general-purpose scripting language that is especially suited for Web development and can be embedded into HTML. The PHP code is executed on the server, generating HTML which is then sent to the client. The client would receive the results of running that script, but would not know what the underlying code was. The user can't know the PHP code, only the result.\[15\]

PHP is used in our application for the main page (index.php) for several purposes and is used also for the communication with the BD (ClearBD and redis).

To collect the value sent by post method in the PHP script we need this code:

```
$_POST["(name of the variable)"];
```

5.6.1. Facebook SDK for PHP

The Facebook SDK for PHP provides a rich set of server-side functionality for accessing Facebook’s server-side API calls. Is used to perform operations as an app administrator.\[16\]

In the application we use this for take the user information, like his name, friends, friends playing, etc.

5.6.2. predis

Predis is a flexible and feature-complete PHP client library for the Redis key-value store. Is used for the communication with the Redis cloud BD.

5.6.3. Alternative to PHP

A part of PHP we can use other programming languages to connect to our databases.

- ClearBD: Ruby and Python/Django. I chose PHP in this case because I did something in php before and is a programming language commonly known.

- Redis cloud: Ruby, Java, Python and Node.js. I did the Clear BD in PHP and I did not think to change the programming language.
5.6.4. Sequel pro

Is a management tool for Mac OS X. This tool is for the management of the SQL databases and is very easy to use it. [18]

To do the connection you only need:

- Host adress
- username no acces
- password of the user
- database name
- port

You can chose the option to connect using SSL. Sequel pro has a very intuitive interface. You can choose the DB in the top-right corner and then you can see the tables on it.

When you choose the table you can change the structure, see and change the content, make relations, add triggers, see the table info and make Query for management.
6. Development and Implementation

The most important things in this project are the connections with Facebook and the data store. All games have been found online and changed later to put them on facebook.

6.1. Start with the app on Facebook.

6.1.1. Create app
To develop a game on Facebook the first thing is to have a Facebook account. For it only has to go to the official website https://www.facebook.com[21] and register on it. Once in our account, we should go to the Facebook developers site https://developers.facebook.com/apps[19] and create new app.

If you have another hosting different to Heroku do not check the last option. If you don't have a Heroku account you can create one in this step.

Now you can edit the information of your app in the next page. You can edit a lot of things like the name, App domains, hosting URL, developer Roles, payments, etc.
6.1.2. Submit app

Once you have finished the game you still needs to do the last step to being able to have the game in the app center: submit the app. To do that you need to go to Facebook developers app page (https://developers.facebook.com/apps\[19\]) and select the app you want to submit and click “edit app”.

In basic settings you can change the name of the app and choose the app domain. If you have been chosen Heroku when you created the app some information has been completed: site URL, canvas URL, appDomain, etc. Also you can add a mobile web site, Native iOS App and Native Android App.

To submit the app you should go to App details.

In the app details you need to complete some sections for submit the app:

- Primary Language: The main language of your app.
- Display name: The app name.
- Description: Short description of your app.
- Detailed description: A good description of your app.
• Privacy Policy URL: You need a privacy policy. When you have a page with it you should put the URL here.

• Icons: You need some icons for your app page.

• Screenshots: Some screenshots of your app.

When you complete this page you can submit your app and automatically creates a page for your application on Facebook.

Example with the connect4:
6.1.3. Heroku

When you create your app on Facebook a template app, in this case written in PHP, has been copied and deployed.

If you go to your Heroku account, in the section apps you will find something like that:

Heroku is a good hosting with addons, good documentation, good support and easy management. Is easy create a new app and add more things in it.
If you chose one of your apps you can see the latest activities, add collaborators to your app, change some settings or add new add-ons or see the add-ons you already have.

This is an example of Connect4 game. This one have two free add-ons: ClearBD and Redis cloud.

Heroku has a lot of add-ons for a lot of things:

- Data store
- Search
- Email and SMS
- Analytics
- Monitoring
- Utilities
- Mobile
- Logging
- Workers and queueing
- Caching
- Media
- Payments

Here is a tutorial explaining how to start an application on heroku:

https://devcenter.heroku.com/articles/facebook
6.2. Snake

I found a tutorial for create a snake game in HTML5 here: http://juegoscanvas.blogspot.co.uk/. I have changed the code to use the Facebook account to store the maximum score.

You can find the game here:

- Facebook page: https://www.facebook.com/appcenter/264925106974559? preview=1&locale=es_ES
- Game on Facebook: https://apps.facebook.com/264925106974559/? fb_source=appcenter&fb_appcenter=1

6.2.1. What is Snake?

Snake is a casual video game that originated during the late 1970s in arcades becoming something of a classic. In 1998 the game appear on Nokia mobile phones and Snake found a massive audience.

In the game, the player controls a thin creature which roams around a plane picking up food (in this case and apple), trying to avoid hitting its own tail. When the snake eats a apple, its tail grows longer, making the game increasingly difficult. The user can change the direction of the snake's head (up, down, left or right). When the snake hit its own tail the game finish. The score is the number of apples the snake can eat before hit its own tail. The player pause the game with the key enter.
Now I will explain the changes and the architecture of the game.

6.2.2. index.php
This is the main page of the project. Facebook give you a index template for your apps and you only should modified it for your needs.

In this case we need to put a canvas object inside the index with width 500 and height 300 and make, call the necessary javaScripts (jquery-1.7.1.min.js and game.js) and put invisible input with the userId who is playing, this last thinks is because we need the userId for store the maximum score inside the javascript code. This id is a Facebook id.

Jquery-1.7.1.min.js is the jquery library.

6.2.3. game.js
This script contains everything related to the game, in this case the snake game.

At the beginning the script take the user from the index.php with the code:

```
user=document.getElementById("nombre").value;
```

The most important function is the function run(). This function have the main loop of the game, the first thing the function does is calculate the state of the game from the inputs and then the function paint the new state.

- This new state depends on two things: the last state and the input.
  - If the key enter is active we have two possibilities:
    - If it is running the game is paused
    - If it is paused the pause is removed.
  - If the direction of the snake is the right or the left and the input is the arrow up or arrow down the snake changes its direction towards this input.
  - If the direction of the snake is up or down and the input is the arrow left or arrow right  is the snake changes its direction towards this input.

When the program finish this part then it move the snake to it direction:
- If in this direction found a apple a new part of the snake is added in the head.
- If in this direction found a other part of his body the game end and the state is
reset.

- If in this direction does not find anything a new part is added in the head and a part is deleted in the tail.

In this script (game.js) I did the functions to call PHP scripts. I add the animation of the apple and the max score in the paint function.

Apple animation

- Paint the max score:

  ```javascript
ctx.fillText('Max.Score: '+maxScore,0,20);
```

  - ‘Max.Score’ is a String
  - MaxScore is a variable with the max score
  - 0 and 20 are the position x,y in the canvas.

- Paint the apple animation:

  ```javascript
ctx.drawImage(iFood,timer%4*10, 0, 10, 10, food.x,food.y, 10, 10);
```

  - iFood is the image.
  - timer%4*20 is the position x. The timer is a variable to know the correct frame.
  - 0 the position y.
  - 10,10 is the size (width, height)

- The PHP calls are for:

  - Register the user in the DB if not exist at the beginning.
  - Get the max score of the user at the beginning.
  - Set the new max score of the user if the last score is better when the user lose the game.
6.2.4. Data base
The connections with the DB had made with PHP script. All the PHP include the file in.inc, this file contain the information of the DB. This information is:
- Server address (variable \$db_server).
- Data base user (variable \$db_user).
- User password (variable \$db_pass).
- DB name (variable \$db_dbase).

With this code the php start the connection with the DB:

```php
mysql_connect($db_server, $db_user, $db_pass);
mysql_select_db($db_dbase);
```

In the data base for the snake I only need one table: User.

In this table I only need two columns: user_id and score. user_id is the ID of the user in Facebook and score is the maximum score obtained for this user.

PHP script:

- regis.php: Input variable: id, this id is the Facebook id of the user. Inserts the user passed as a parameter if it does not exist in the DB, the maximum score is 0.
- score.php: Input variable: id, this id is the Facebook id of the user. Returns the user punctuation passed as a parameter.
- setScore.php: Input variable: id and msScore. id is the Facebook id of the user and msScore is the new maximum score of the user. Sets the maximum score received as parameter with the specified user.
6.3. Connect 4

The code of the connect 4 is in: https://github.com/unixpickle/Connect4\(^{[24]}\). This game is only to play in one computer. I have changed the code and now two friends on Facebook can play together with his Facebook account.

I do two version with different DBs:

- **ClearBD SQL**
  - you can find the game here: https://protected-wildwood-8266.herokuapp.com/
  - Facebook page: https://www.facebook.com/appcenter/443501612396104/?preview=1&locale=es_ES
  - Facebook game: https://apps.facebook.com/443501612396104/?fb_source=appcenter&fb_appcenter=1
- **Redis cloud (noSQL)**
  - you can find the game here: https://protected-wildwood-8266.herokuapp.com/indexnosql.php

6.3.1 What is connect4?

This game have a lot of different names: Captain's Mistress, Four Up, Plot Four, Find Four, Fourplay, Four in a row and Four in a line. It is a two-player game, each player have a colour and then they drop a coloured disc from the top and the disc fall straight down occupying the next available space within the column. The objective of the game is to connect four of own discs next to each other vertically, horizontally or diagonally before your opponent. There are many variation of the board size, in my case the size is 7x6.

When the user create a connect4 game, he is the red player and his friend is the blue player. The user who create the game starts and he can click one column, putting a disc in the lowest Y position of this columns. When the user puts a piece on the board, he have to wait for the other person to place a piece.
Now I will explain the changes and the architecture of the game.

6.3.2. index.php
This is the main page of the project. Facebook give you a index template for your apps and you only should modified it for your needs.

In this case we need to change a lot of thinks:

-First the php gets the parameter “a” passed with the method get (in the URL). This parameter only exists if the user start a game with a friend. The parameter “a” is the ID of a friend to play with him.

-If the variable “a” exists then the canvas is added and the user can play with the friend who’s id is “a”. We have two canvas: the canvas for play and the canvas which shows the turn (red or blue).
When the game is created all the javascript are called and start to run. We need five parameters for the game:

- state: the state of the game.
- id1: user id.
- id2: friend id.
- turn: current turn (red or blue).
- jt: turn of the user (red or blue).

These parameters are input hidden in the php, it's the easy way to take a parameter from the javascript.

Example:

Inside the php the program has a input with id “turn” and the attribute “hidden” with the same value as $turn:

```html
<input type="hidden" id="turn" value=<?= $turn ?> />
```

This value can be got in the javascript with: `document.getElementById("turn").value;`

-The php add a section to continue the game with a friend pressing the button “Game”. The games shown below are the games in the DB with the user friends. When the user press the button, this button call a javascript function to recharge the page with the parameter “a”, this parameter is the userID of the friend the user want to play.
- The php add a section to create a game with the user friends who are playing this game but the user don't have a game with them. To do that the user can press the button “invite”. This button add a game in the DB between the user and his friend. When the user press the button, this button call a javascript function to prepare the call to the PHP to create a new row in the DB.

### 6.3.3. GameEngine.js
This script contains the global variables and is the first script called when all is loaded.

The main functionalities are:

- configure the canvas.
- create the game board.
- paints for the first time the board.

Here I add the call to get the second canvas. This canvas is for paint the player turn on it.

### 6.3.4. GameBoard.js
This script have three functions:

- Constructor function. When the program create a GameBoard this script initialize the variables and create the GameState, this game state needs the size of the board, in this case 7x6.

- When the user click in the canvas the function MouseDown is called.

  If it is not the player turn this function does nothing but if is the player turn and the variable “winner” is equal to 0 (nobody wins) this function put the new piece in the board and update the state in the DB. When you click in a column the piece fall at the less position Y it can, if the column is full the movement is invalid.
If the variable “winner” is different to 0 this means one player won the game. Then this function increment the wins of the player who won the game.

- When the player make a movement and the state change we need to paint again the canvas. In this script we have the function draw which is responsible for managing the painting. This function paint the background, the separators between squares and have the loop to paint the pieces. If the variable winner is different to 0 the winner dialog is painted. If is not the turn of this player and nobody won the game the function “checkTurn” is called for wait him turn.

### 6.3.5. GameState.js

This script contains the state of the game. When the program do a new GameState this take some variables from the php, this variables are:

- state: the state of the game.
- id1: user id.
- id2: friend id.
- turn: current turn (1 if it is the red player or 2 if it is the blue player).
- jt: turn of the user (1 if it is the red player or 2 if it is the blue player).

Also initialize some variables for the game.

Here we have the polling to the DB to know if is the player turn or not in the function “checkTurn”. This polling is 300 milliseconds+ResponseTime. To know if is the player turn we only need to compare if the turn stored in the DB is the same number as the player. When the program know is the player turn its get the state from the DB to actualize the state.

We have also functions in this script to call the php to save the game. We have two functions for one reason: in one of them the script change the turn but we do not need that if we want to store the initial state because we already have the correct turn. To save the game we need to pass the array to a string, I do this with a loop to pass the entire array to a string.

Here we have the function to add a new piece in the board and a function to check if one
player won the game in the last movement or not. To do that the script check if some one have four pieces in a row, there are four possibilities:
6.3.6. Game State

The game state have two things:

- Board
- Turn

In the program the board is a array called piecesG, this array contains integers. The size of this array is 7x6, the same as the board size. The position of the board in the array is \(x+y\times7\).

<table>
<thead>
<tr>
<th>Array pos 0+0\times7=0</th>
<th>Array pos 1+0\times7=1</th>
<th>Array pos 2+0\times7=2</th>
<th>Array pos 3+0\times7=3</th>
<th>Array pos 4+0\times7=4</th>
<th>Array pos 5+0\times7=5</th>
<th>Array pos 6+0\times7=6</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Array pos 0+1\times7=7</td>
<td>Array pos 1+1\times7=8</td>
<td>Array pos 2+1\times7=9</td>
<td>Array pos 3+1\times7=10</td>
<td>Array pos 4+1\times7=11</td>
<td>Array pos 5+1\times7=12</td>
<td>Array pos 6+1\times7=13</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Array pos 0+2\times7=14</td>
<td>Array pos 1+2\times7=15</td>
<td>Array pos 2+2\times7=16</td>
<td>Array pos 3+2\times7=17</td>
<td>Array pos 4+2\times7=18</td>
<td>Array pos 5+2\times7=19</td>
<td>Array pos 6+2\times7=20</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Array pos 0+3\times7=21</td>
<td>Array pos 1+3\times7=22</td>
<td>Array pos 2+3\times7=23</td>
<td>Array pos 3+3\times7=24</td>
<td>Array pos 4+3\times7=25</td>
<td>Array pos 5+3\times7=26</td>
<td>Array pos 6+3\times7=27</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Array pos 0+4\times7=28</td>
<td>Array pos 1+4\times7=29</td>
<td>Array pos 2+4\times7=30</td>
<td>Array pos 3+4\times7=31</td>
<td>Array pos 4+4\times7=32</td>
<td>Array pos 5+4\times7=33</td>
<td>Array pos 6+4\times7=34</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Array pos 0+5\times7=35</td>
<td>Array pos 1+5\times7=36</td>
<td>Array pos 2+5\times7=37</td>
<td>Array pos 3+5\times7=38</td>
<td>Array pos 4+5\times7=39</td>
<td>Array pos 5+5\times7=40</td>
<td>Array pos 6+5\times7=41</td>
</tr>
</tbody>
</table>

The integers inside the array can be:

- 0 : the position is empty.
- 1 : the position have a red piece from the player 1.
- 2 : the position have a blue piece from the player 2.

The variable turn is an integer, the integer only can be two values:

- 1 : is the turn of the player red.
- 2: is the turn of the player blue.
6.3.7. Data base

The connections with the DB had made with PHP. In this game I implemented two DBs, the first DB use SQL (ClearBD) and the second one do not use SQL (Redis cloud).

The information in the DB is the same in both cases:

- **id1**: Facebook id of the player red.
- **id2**: Facebook id of the player blue.
- **state**: Board state. This is the string with the information of the array state mentioned before. This string contains 0,1 and 2 like the board matrix.
- **turn**: This is the current turn. 1 if it is red turn and 2 if it is blue turn.
- **p1**: How many wins have the red player.
- **p2**: How many wins have the blue player.
- **aux**: When someone win a game the two players try to increment the number of win of the winner, to prevent a double increment I have this variable. When this variable is equal to 0 this PHP increment the number of wins of the winner and this variable becomes 1. If this variable is equal to 1 this PHP do not increment the winnings and becomes 0.

The primary key of this DB is Id1+Id2 and Id1+Id2 is the same than Id2+Id1, that means in my BD can't exists two games between the same two players. Both of them use the same PHP.

- **existGame.php**: Input variables: Id1 and Id2. Id1 is the Facebook id of the user who is calling the PHP and Id2 is the Facebook id of the friend. This PHP return 1 in case exists a game between the two Ids passed as parameters. Returns 0 otherwise.
- **getState.php**: Input variables: Id1 and Id2. Id1 is the Facebook id of the user who is calling the PHP and Id2 is the Facebook if of the friend. This PHP return the state of the game between id1 and id2.
- **getTurn.php**: Input variables: Id1 and Id2. Id1 is the Facebook id of the user who is calling the PHP and Id2 is the Facebook id of the friend. This PHP return the turn of the game between id1 and id2.
• **newGame.php:** Input variables: Id1 and Id2. Id1 is the Facebook id of the user who is calling the PHP and Id2 is the Facebook id of the friend. This PHP create a new game in the database. And initialize the information of the game:

  state = '0000000000000000000000000000000000000000000'. Empty board.

  turn = 1. The player red always is the first.

  p1 = 0. The player red have 0 wins at the beginning.

  p2 = 0. The player blue have 0 wins at the beginning.

  aux = 0. This must be 0 because we can increment the winner count.

• **saveSinc.php:** Input variables: w, Id1 and Id2. w is the variable for determinate who won the game, if this variable is 1 the winner is the red player, if it is 2 the winner is the blue player. If aux = 0 this php change aux to 1 and increment the winner count of the winner, if the w is equal to 1 the php increment p1, if w is equal to 2 this script increment p2. If aux = 1 this php only change aux to 0. Id1 is the Facebook id of the user who is calling the PHP and Id2 is the Facebook id of the friend.

• **saveState.php:** Input variables: Id1, Id2, s and t. Id1 is the Facebook id of the user who is calling the PHP and Id2 is the Facebook id of the friend. s is the state of the game and t the current turn. This php store the state and the turn between the player id1 and the player id2.

6.3.8. ClearBD SQL

All the PHP to connect with ClearBD SQL include the file in1.inc, this file contain the information of the DB like in the snake with the same structure to access to the DB:

- Server address (variable $db_server).

- Data base user (variable $db_user).

- User password (variable $db_pass).

- DB name (variable $db_dbase).
With this code the PHP starts the connection with the DB:

```php
mysql_connect($db_server, $db_user, $db_pass);
mysqlSelect_db($db_dbase);
```

With this the PHP can do the queries as normal.

```php
mysql_query("(query)");
```

To know more about this more information here: [http://php.net/][25].

### 6.3.9. Redis cloud (noSQL)

Redis works different than a SQL DB, its needs a key to store the information and with this key you can access the information related with this key. To store the information I used the hash data type, in this case the key is id1+id2, where id1 is the id of the user who makes the game and id2 is the id of the friend. To not have two games between the same player we need to check id1+id2 and id2+id1 because the game between id1 and id2 is the same as have a game between id2+id1. With PHP and the key we can get the game information (state, turn, p1, p2 and aux).

All the PHP have this code:

```php
require 'predis-0.8/autoload.php';

Predis\Autoloader::register();

$redis = new Predis\Client(array(
    'host' => parse_url($_ENV['REDISCLOUD_URL'], PHP_URL_HOST),
    'port' => parse_url($_ENV['REDISCLOUD_URL'], PHP_URL_PORT),
    'password' => parse_url($_ENV['REDISCLOUD_URL'], PHP_URL_PASS),
));
```

With this code you can access to the Redis cloud stored in the Heroku app. We need to add the first line to load the PHP code to work with Redis in PHP.

To know more about how to use Redis in PHP: [http://phpmaster.com/an-introduction-to-redis-in-php-using-predis/][20].
6.4. MMOG (Massive Multiplayer Online Game)

I have a single-player game and two-player game, then the next logical step is a MMO game for Facebook. I did not implement this one because I haven't enough time to do it. This kind of game have a lot of classes inside and store a lot information, that makes it difficult and needs more time to be implemented.

At the moment, the single player did not need communication between players because they play alone, in the two-player game, in this case the connect4 I do the connection with a polling in the DB, I can do that because is a turn-base game and it is asynchronous. Players take their time, contemplating the possibilities and they can do the movement when they want. In the browser is more difficult to do a real time game than a turn-based game.

To do a MMOG we need to do a real time connection and HTTPs is not enough to do that alone because the real time game require messages send and received sometimes in the region of 33-66 times per second. Luckily, in modern browsers we can take one step higher, and have a real time connection between a server and clients. The purpose of this discussion is to present one overview of how multiplayer games are made.\[26\]

I found a MMOG with a open-source code in internet, its name is Cobalt Calibu, you can found it in [https://www.openshift.com/blogs/hosting-and-developing-the-html5-game-cobalt-calibur-free-on-openshift](https://www.openshift.com/blogs/hosting-and-developing-the-html5-game-cobalt-calibur-free-on-openshift).\[27\]

The main idea was take a MMOG like that and make changes to put it on Facebook. The way to do that is remove the registration and the login and put a Facebook login to save the game status. With that kind of game is a good idea make a market for the player and sell some game items to earn money like a lot Facebook games.
7. Testing
To see some bugs inside the games and test the performance I tested the games with some friends.

7.1. Snake
This game is very simple and only access to the DB when the game is over and the user do a better score than he did before.

I tested this game with nine people playing at the same time and it works correctly. Is not a lot of people but I this it is ok with this free version of the ClearDB SQL.

In this game I didn't find any bug.

7.2. Connect 4
Twelve friends play that game to test it and I found some bugs while we were testing it.

- When a player win a game, his victory count is incremented by 2. This bug happens because when a player win a game the two players increment the count. Is easy to fix this bug using a aux variable, when it is 0 the user increment the count and put this variable = 1, when this variable is 1 the user don't increment the count and put this variable = 0.

- If one player open two windows and make a movement in one of them the state is stored and the game continues normally but when makes a movement in the other windows this state is stored too, cancelling the others movements. This bug is easy to solve with a checksum in the php which save the game. Before store the new state it check this: number of pieces in the old state+1 = number of pieces in the new state, if this is true the php store the new state.

  - With this the php don't store a new game because always n+1 > 0 where n is a positive integer. For solve this the php check if the number of pieces in the new states is equal to 0.

- When a player win a game if the two players close the windows and open it again the dialog with a winner don't appear and the player can do a movement. This is because at the beginning the game don't check if someone won the game. To solve this the program only should check if someone won the game when the game is loaded.
I implemented two DBs for this game. One of them with SQL (ClearBD) and other one without SQL (redis cloud). I do tests in different conditions. The time is in milliseconds, collected by the javascript function Date.now(), I capture the time when a player send the state and when the other player receive that state.

Always the polling is taking in to account the response time.

7.2.1. SQL tests
• Nine games at the same time (polling = 300 milliseconds+response):

<table>
<thead>
<tr>
<th>Player 1 (chrome)</th>
<th>Player 2 (safari)</th>
<th>Results</th>
</tr>
</thead>
<tbody>
<tr>
<td>Send: 1367999506881</td>
<td>Send: 1367999506416</td>
<td>465</td>
</tr>
<tr>
<td>Receive: 1367999527146</td>
<td>Receive: 1367999527571</td>
<td>425</td>
</tr>
<tr>
<td>Send: 1367999528965</td>
<td>Send: 1367999528462</td>
<td>503</td>
</tr>
<tr>
<td>Receive: 1367999529853</td>
<td>Receive: 1367999530502</td>
<td>649</td>
</tr>
<tr>
<td>Send: 1367999532005</td>
<td>Send: 1367999531681</td>
<td>324</td>
</tr>
<tr>
<td>Receive: 1367999532745</td>
<td>Receive: 1367999533502</td>
<td>757</td>
</tr>
<tr>
<td>Send: 1367999534709</td>
<td>Send: 1367999534191</td>
<td>518</td>
</tr>
<tr>
<td>Receive: 1367999535668</td>
<td>Receive: 1367999535941</td>
<td>273</td>
</tr>
<tr>
<td>Send: 1367999537772</td>
<td>Send: 1367999537075</td>
<td>697</td>
</tr>
<tr>
<td>Receive: 1367999538728</td>
<td>Receive: 1367999538913</td>
<td>185</td>
</tr>
<tr>
<td>Send: 1367999540516</td>
<td>Send: 1367999539807</td>
<td>709</td>
</tr>
<tr>
<td>Receive: 1367999541524</td>
<td>Receive: 1367999542046</td>
<td>522</td>
</tr>
<tr>
<td>Send: 1367999542758</td>
<td>Send: 1367999542500</td>
<td>258</td>
</tr>
<tr>
<td>Receive: 1367999544498</td>
<td>Receive: 1367999544854</td>
<td>356</td>
</tr>
<tr>
<td>Send: 1367999548350</td>
<td>Send: 1367999547818</td>
<td>532</td>
</tr>
<tr>
<td>Receive: 1367999554340</td>
<td>Receive: 1367999549707</td>
<td>367</td>
</tr>
<tr>
<td>Receive:</td>
<td>Send:</td>
<td>Results</td>
</tr>
<tr>
<td>----------</td>
<td>------------</td>
<td>---------</td>
</tr>
<tr>
<td>1367599553711</td>
<td>1367599552982</td>
<td>729</td>
</tr>
<tr>
<td>Send:</td>
<td>Receive:</td>
<td></td>
</tr>
<tr>
<td>1367599554424</td>
<td>1367599554880</td>
<td>456</td>
</tr>
<tr>
<td>Receive:</td>
<td>Send:</td>
<td></td>
</tr>
<tr>
<td>1367599556608</td>
<td>1367599556110</td>
<td>498</td>
</tr>
<tr>
<td>Send:</td>
<td>Receive:</td>
<td></td>
</tr>
<tr>
<td>1367599557993</td>
<td>1367599558328</td>
<td>335</td>
</tr>
<tr>
<td>Receive:</td>
<td>Send:</td>
<td></td>
</tr>
<tr>
<td>1367599559784</td>
<td>1367599559221</td>
<td>563</td>
</tr>
</tbody>
</table>

Average: 440.04 milliseconds

- Only one game running (polling = 300 milliseconds+response):

<table>
<thead>
<tr>
<th>Player 1 (chrome)</th>
<th>Player 2 (safari)</th>
<th>Results</th>
</tr>
</thead>
<tbody>
<tr>
<td>Receive:</td>
<td>Send:</td>
<td></td>
</tr>
<tr>
<td>1367582799496</td>
<td>1367582799144</td>
<td>352</td>
</tr>
<tr>
<td>Send:</td>
<td>Receive:</td>
<td></td>
</tr>
<tr>
<td>1367582800240</td>
<td>1367582800478</td>
<td>238</td>
</tr>
<tr>
<td>Receive:</td>
<td>Send:</td>
<td></td>
</tr>
<tr>
<td>1367582801974</td>
<td>1367582801726</td>
<td>248</td>
</tr>
<tr>
<td>Send:</td>
<td>Receive:</td>
<td></td>
</tr>
<tr>
<td>1367582802694</td>
<td>1367582803063</td>
<td>369</td>
</tr>
<tr>
<td>Receive:</td>
<td>Send:</td>
<td></td>
</tr>
<tr>
<td>1367582804467</td>
<td>1367582804235</td>
<td>232</td>
</tr>
<tr>
<td>Send:</td>
<td>Receive:</td>
<td></td>
</tr>
<tr>
<td>1367582805576</td>
<td>1367582805890</td>
<td>314</td>
</tr>
<tr>
<td>Receive:</td>
<td>Send:</td>
<td></td>
</tr>
<tr>
<td>1367582808442</td>
<td>1367582806890</td>
<td>1552</td>
</tr>
<tr>
<td>Send:</td>
<td>Receive:</td>
<td></td>
</tr>
<tr>
<td>1367582810151</td>
<td>1367582810843</td>
<td>892</td>
</tr>
<tr>
<td>Receive:</td>
<td>Send:</td>
<td></td>
</tr>
<tr>
<td>1367582812000</td>
<td>1367582811547</td>
<td>453</td>
</tr>
<tr>
<td>Send:</td>
<td>Receive:</td>
<td></td>
</tr>
<tr>
<td>1367582813214</td>
<td>1367582813719</td>
<td>505</td>
</tr>
<tr>
<td>Receive:</td>
<td>Send:</td>
<td></td>
</tr>
<tr>
<td>1367582815387</td>
<td>1367582814945</td>
<td>442</td>
</tr>
<tr>
<td>Send:</td>
<td>Receive:</td>
<td></td>
</tr>
<tr>
<td>1367582829900</td>
<td>1367582830667</td>
<td>767</td>
</tr>
<tr>
<td>Receive:</td>
<td>Send:</td>
<td></td>
</tr>
<tr>
<td>1367582831576</td>
<td>1367582831225</td>
<td>351</td>
</tr>
<tr>
<td>Send:</td>
<td>Receive:</td>
<td></td>
</tr>
<tr>
<td>1367582832293</td>
<td>1367582832541</td>
<td>248</td>
</tr>
<tr>
<td>Receive:</td>
<td>Send:</td>
<td></td>
</tr>
<tr>
<td>1367582833937</td>
<td>1367582833636</td>
<td>301</td>
</tr>
</tbody>
</table>
Average: 447.59 milliseconds

• Nine games at the same time (polling = 0 milliseconds+response):

<table>
<thead>
<tr>
<th>Send: 1367582835697</th>
<th>Receive: 1367582835895</th>
<th>198</th>
</tr>
</thead>
<tbody>
<tr>
<td>Receive: 1367582837569</td>
<td>Send: 1367582837393</td>
<td>176</td>
</tr>
<tr>
<td>Send: 1367582838287</td>
<td>Receive: 1367582838893</td>
<td>806</td>
</tr>
<tr>
<td>Receive: 1367582840229</td>
<td>Send: 1367582839670</td>
<td>559</td>
</tr>
<tr>
<td>Send: 1367582841063</td>
<td>Receive: 1367582841611</td>
<td>548</td>
</tr>
<tr>
<td>Receive: 1367582842545</td>
<td>Send: 1367582842369</td>
<td>176</td>
</tr>
<tr>
<td>Send: 1367582844130</td>
<td>Receive: 1367582844650</td>
<td>520</td>
</tr>
</tbody>
</table>

Player 1 (chrome)                  | Player 2 (safari)                  | Results |
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Receive: 1367584441696</td>
<td>Send: 1367584440988</td>
<td>708</td>
</tr>
<tr>
<td>Send: 1367584443168</td>
<td>Receive: 1367584443512</td>
<td>344</td>
</tr>
<tr>
<td>Receive: 1367584446096</td>
<td>Send: 1367584445829</td>
<td>267</td>
</tr>
<tr>
<td>Send: 1367584447485</td>
<td>Receive: 1367584447759</td>
<td>274</td>
</tr>
<tr>
<td>Receive: 1367584449922</td>
<td>Send: 1367584449139</td>
<td>783</td>
</tr>
<tr>
<td>Send: 1367584451255</td>
<td>Receive: 1367584451575</td>
<td>320</td>
</tr>
<tr>
<td>Receive: 1367584453535</td>
<td>Send: 1367584452943</td>
<td>592</td>
</tr>
<tr>
<td>Send: 1367584457690</td>
<td>Receive: 1367584458002</td>
<td>312</td>
</tr>
<tr>
<td>Receive: 1367584459047</td>
<td>Send: 1367584458748</td>
<td>299</td>
</tr>
<tr>
<td>Send: 1367584460795</td>
<td>Receive: 1367584461312</td>
<td>517</td>
</tr>
<tr>
<td>Receive: 1367584497836</td>
<td>Send: 1367584497335</td>
<td>501</td>
</tr>
<tr>
<td>Send: 1367584498902</td>
<td>Receive: 1367584499317</td>
<td>415</td>
</tr>
<tr>
<td>Receive: 1367584500801</td>
<td>Send: 1367584500538</td>
<td>263</td>
</tr>
<tr>
<td>Send: 1367584501762</td>
<td>Receive: 1367584502132</td>
<td>370</td>
</tr>
<tr>
<td>Receive: 1367584503796</td>
<td>Send: 1367584503441</td>
<td>355</td>
</tr>
</tbody>
</table>
Average: 421.19 milliseconds

- Only one game running (polling = 0 milliseconds+response):

<table>
<thead>
<tr>
<th>Send:</th>
<th>Receive:</th>
</tr>
</thead>
<tbody>
<tr>
<td>1367584504836</td>
<td>1367584505138</td>
</tr>
<tr>
<td>1367584506756</td>
<td>1367584506417</td>
</tr>
<tr>
<td>1367584508106</td>
<td>1367584508289</td>
</tr>
<tr>
<td>1367584510556</td>
<td>1367584510304</td>
</tr>
<tr>
<td>1367584512024</td>
<td>1367584512653</td>
</tr>
<tr>
<td>1367584512284</td>
<td>1367584513104</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Player 1 (chrome)</th>
<th>Player 2 (safari)</th>
<th>Results</th>
</tr>
</thead>
<tbody>
<tr>
<td>Receive: 1367585010544</td>
<td>Send: 1367585010293</td>
<td>251</td>
</tr>
<tr>
<td>Send: 1367585012660</td>
<td>Receive: 1367585012889</td>
<td>229</td>
</tr>
<tr>
<td>Receive: 1367585014617</td>
<td>Send: 1367585014466</td>
<td>151</td>
</tr>
<tr>
<td>Send: 1367585015740</td>
<td>Receive: 1367585015977</td>
<td>237</td>
</tr>
<tr>
<td>Receive: 1367585017520</td>
<td>Send: 1367585017234</td>
<td>286</td>
</tr>
<tr>
<td>Send: 1367585018500</td>
<td>Receive: 1367585018702</td>
<td>202</td>
</tr>
<tr>
<td>Receive: 1367585024326</td>
<td>Send: 1367585023846</td>
<td>480</td>
</tr>
<tr>
<td>Send: 1367585025263</td>
<td>Receive: 1367585025439</td>
<td>176</td>
</tr>
<tr>
<td>Receive: 1367585026892</td>
<td>Send: 1367585026900</td>
<td>8</td>
</tr>
<tr>
<td>Send: 1367585028026</td>
<td>Receive: 1367585028563</td>
<td>537</td>
</tr>
<tr>
<td>Receive: 1367585029693</td>
<td>Send: 1367585029490</td>
<td>203</td>
</tr>
<tr>
<td>Send: 1367585030946</td>
<td>Receive: 1367585031481</td>
<td>535</td>
</tr>
<tr>
<td>Receive: 1367585032877</td>
<td>Send: 1367585032686</td>
<td>191</td>
</tr>
<tr>
<td>Send: 1367585034014</td>
<td>Receive: 1367585034436</td>
<td>422</td>
</tr>
<tr>
<td>Receive: 1367585035634</td>
<td>Send: 1367585035268</td>
<td>366</td>
</tr>
<tr>
<td>Send:</td>
<td>Receive:</td>
<td>Time</td>
</tr>
<tr>
<td>-----------</td>
<td>-----------------</td>
<td>-------</td>
</tr>
<tr>
<td>1367585036550</td>
<td>1367585036875</td>
<td>325</td>
</tr>
<tr>
<td>1367585038817</td>
<td>1367585037949</td>
<td>868</td>
</tr>
<tr>
<td>1367585039808</td>
<td>1367585040288</td>
<td>480</td>
</tr>
<tr>
<td>1367585041252</td>
<td>1367585041052</td>
<td>200</td>
</tr>
<tr>
<td>1367585046261</td>
<td>1367585046701</td>
<td>440</td>
</tr>
<tr>
<td>1367585051934</td>
<td>1367585051768</td>
<td>166</td>
</tr>
<tr>
<td>1367585053950</td>
<td>1367585054123</td>
<td>173</td>
</tr>
</tbody>
</table>

Average: 314.82 milliseconds
7.2.2. noSQL tests

- Nine games at the same time (polling = 300 milliseconds+response):

<table>
<thead>
<tr>
<th>Player 1 (chrome)</th>
<th>Player 2 (safari)</th>
<th>Results</th>
</tr>
</thead>
<tbody>
<tr>
<td>1367596174857</td>
<td>1367596175293</td>
<td>436</td>
</tr>
<tr>
<td>1367596176920</td>
<td>1367596176571</td>
<td>349</td>
</tr>
<tr>
<td>1367596178070</td>
<td>1367596178593</td>
<td>523</td>
</tr>
<tr>
<td>1367596190700</td>
<td>1367596190508</td>
<td>192</td>
</tr>
<tr>
<td>1367596192427</td>
<td>1367596192663</td>
<td>236</td>
</tr>
<tr>
<td>1367596194134</td>
<td>1367596193733</td>
<td>401</td>
</tr>
<tr>
<td>1367596194830</td>
<td>1367596194984</td>
<td>154</td>
</tr>
<tr>
<td>1367596196373</td>
<td>1367596195787</td>
<td>586</td>
</tr>
<tr>
<td>1367596197537</td>
<td>1367596198007</td>
<td>470</td>
</tr>
<tr>
<td>1367596200196</td>
<td>1367596200014</td>
<td>182</td>
</tr>
<tr>
<td>1367596201998</td>
<td>1367596202335</td>
<td>337</td>
</tr>
<tr>
<td>1367596203905</td>
<td>1367596203351</td>
<td>554</td>
</tr>
<tr>
<td>1367596204779</td>
<td>1367596205028</td>
<td>249</td>
</tr>
<tr>
<td>1367596211140</td>
<td>1367596211615</td>
<td>325</td>
</tr>
<tr>
<td>1367596214471</td>
<td>1367596214943</td>
<td>472</td>
</tr>
<tr>
<td>1367596216219</td>
<td>1367596215738</td>
<td>481</td>
</tr>
<tr>
<td>1367596217225</td>
<td>1367596217499</td>
<td>274</td>
</tr>
<tr>
<td>1367596219546</td>
<td>1367596218683</td>
<td>863</td>
</tr>
<tr>
<td>1367596220472</td>
<td>1367596220756</td>
<td>284</td>
</tr>
<tr>
<td>1367596222233</td>
<td>1367596221946</td>
<td>287</td>
</tr>
<tr>
<td>1367596233915</td>
<td>1367596234058</td>
<td>143</td>
</tr>
<tr>
<td>1367596235660</td>
<td>1367596235431</td>
<td>229</td>
</tr>
<tr>
<td>1367596237129</td>
<td>1367596237573</td>
<td>444</td>
</tr>
</tbody>
</table>

Average: 368.3 milliseconds
Only one game running (polling = 300 milliseconds+response):

<table>
<thead>
<tr>
<th>Player 1 (chrome)</th>
<th>Player 2 (safari)</th>
<th>Results</th>
</tr>
</thead>
<tbody>
<tr>
<td>Send: 1367594950069</td>
<td>Receive: 1367594950951</td>
<td>882</td>
</tr>
<tr>
<td>Receive: 1367595500164</td>
<td>Send: 13675955499942</td>
<td>222</td>
</tr>
<tr>
<td>Send: 1367595501165</td>
<td>Receive: 1367595501611</td>
<td>446</td>
</tr>
<tr>
<td>Send: 1367595503598</td>
<td>Receive: 1367595503431</td>
<td>167</td>
</tr>
<tr>
<td>Send: 1367595505246</td>
<td>Receive: 1367595505423</td>
<td>177</td>
</tr>
<tr>
<td>Send: 1367595508605</td>
<td>Receive: 1367595508182</td>
<td>423</td>
</tr>
<tr>
<td>Send: 1367595509965</td>
<td>Receive: 1367595510230</td>
<td>265</td>
</tr>
<tr>
<td>Send: 1367595512261</td>
<td>Receive: 1367595511688</td>
<td>573</td>
</tr>
<tr>
<td>Send: 1367595513318</td>
<td>Receive: 1367595513804</td>
<td>486</td>
</tr>
<tr>
<td>Send: 1367595516786</td>
<td>Receive: 1367595516355</td>
<td>431</td>
</tr>
<tr>
<td>Send: 1367595517999</td>
<td>Receive: 1367595518340</td>
<td>341</td>
</tr>
<tr>
<td>Send: 1367595520236</td>
<td>Receive: 1367595520062</td>
<td>174</td>
</tr>
<tr>
<td>Send: 1367595521465</td>
<td>Receive: 1367595521670</td>
<td>205</td>
</tr>
<tr>
<td>Send: 1367595523478</td>
<td>Receive: 1367595522944</td>
<td>534</td>
</tr>
<tr>
<td>Send: 1367595524383</td>
<td>Receive: 1367595524688</td>
<td>305</td>
</tr>
<tr>
<td>Send: 1367595526617</td>
<td>Receive: 1367595526389</td>
<td>228</td>
</tr>
<tr>
<td>Send: 1367595527809</td>
<td>Receive: 1367595528133</td>
<td>324</td>
</tr>
<tr>
<td>Send: 1367595529706</td>
<td>Receive: 1367595529498</td>
<td>208</td>
</tr>
<tr>
<td>Send: 1367595531158</td>
<td>Receive: 1367595531483</td>
<td>325</td>
</tr>
<tr>
<td>Send: 1367595533284</td>
<td>Receive: 1367595533153</td>
<td>131</td>
</tr>
<tr>
<td>Send: 1367595534070</td>
<td>Receive: 1367595534207</td>
<td>137</td>
</tr>
<tr>
<td>Send: 1367595536859</td>
<td>Receive: 1367595536710</td>
<td>149</td>
</tr>
<tr>
<td>Send: 1367595538129</td>
<td>Receive: 1367595538724</td>
<td>595</td>
</tr>
<tr>
<td>Send: 1367595540309</td>
<td>Receive: 1367595539836</td>
<td>473</td>
</tr>
</tbody>
</table>

Average: 341.7 milliseconds
Nine games at the same time (polling = 0 milliseconds+response):

<table>
<thead>
<tr>
<th>Player 1 (chrome)</th>
<th>Player 2 (safari)</th>
<th>Results</th>
</tr>
</thead>
<tbody>
<tr>
<td>Send: 1367597399322</td>
<td>Receive: 1367597399472</td>
<td>150</td>
</tr>
<tr>
<td>Receive: 1367597403591</td>
<td>Send: 1367597403386</td>
<td>205</td>
</tr>
<tr>
<td>Send: 1367597404576</td>
<td>Receive: 1367597404791</td>
<td>215</td>
</tr>
<tr>
<td>Receive: 1367597405974</td>
<td>Send: 1367597405792</td>
<td>182</td>
</tr>
<tr>
<td>Send: 1367597406712</td>
<td>Receive: 1367597406936</td>
<td>224</td>
</tr>
<tr>
<td>Receive: 1367597407944</td>
<td>Send: 1367597407693</td>
<td>251</td>
</tr>
<tr>
<td>Send: 1367597408885</td>
<td>Receive: 1367597409117</td>
<td>232</td>
</tr>
<tr>
<td>Receive: 1367597409977</td>
<td>Send: 1367597409717</td>
<td>260</td>
</tr>
<tr>
<td>Send: 1367597410726</td>
<td>Receive: 1367597410909</td>
<td>183</td>
</tr>
<tr>
<td>Receive: 1367597411756</td>
<td>Send: 1367597411523</td>
<td>233</td>
</tr>
<tr>
<td>Send: 1367597412336</td>
<td>Receive: 1367597412483</td>
<td>147</td>
</tr>
<tr>
<td>Receive: 1367597413659</td>
<td>Send: 1367597413428</td>
<td>231</td>
</tr>
<tr>
<td>Send: 1367597414666</td>
<td>Receive: 1367597414889</td>
<td>223</td>
</tr>
<tr>
<td>Receive: 1367597418285</td>
<td>Send: 1367597418138</td>
<td>147</td>
</tr>
<tr>
<td>Send: 1367597419286</td>
<td>Receive: 1367597419501</td>
<td>215</td>
</tr>
<tr>
<td>Receive: 1367597420091</td>
<td>Send: 1367597419928</td>
<td>163</td>
</tr>
<tr>
<td>Send: 1367597421066</td>
<td>Receive: 1367597421262</td>
<td>196</td>
</tr>
<tr>
<td>Receive: 1367597426021</td>
<td>Send: 1367597425749</td>
<td>272</td>
</tr>
<tr>
<td>Send: 1367597427138</td>
<td>Receive: 1367597427342</td>
<td>204</td>
</tr>
<tr>
<td>Receive: 1367597431216</td>
<td>Send: 1367597430917</td>
<td>299</td>
</tr>
<tr>
<td>Send: 1367597433675</td>
<td>Receive: 1367597433841</td>
<td>166</td>
</tr>
</tbody>
</table>

Average: 209.43 milliseconds
• Only one game running (polling = 0 milliseconds+response):

<table>
<thead>
<tr>
<th>Player 1 (chrome)</th>
<th>Player 2 (safari)</th>
<th>Results</th>
</tr>
</thead>
<tbody>
<tr>
<td>Receive : 1367596948613</td>
<td>Send: 1367596948416</td>
<td>197</td>
</tr>
<tr>
<td>Send: 1367596955469</td>
<td>Receive : 1367596955646</td>
<td>177</td>
</tr>
<tr>
<td>Receive : 1367596956878</td>
<td>Send: 1367596956636</td>
<td>242</td>
</tr>
<tr>
<td>Send: 1367596957897</td>
<td>Receive : 1367596958113</td>
<td>216</td>
</tr>
<tr>
<td>Receive : 1367596959382</td>
<td>Send: 1367596959181</td>
<td>201</td>
</tr>
<tr>
<td>Send: 1367596960458</td>
<td>Receive : 1367596960686</td>
<td>228</td>
</tr>
<tr>
<td>Receive : 1367596962223</td>
<td>Send: 1367596962007</td>
<td>216</td>
</tr>
<tr>
<td>Send: 1367596963669</td>
<td>Receive : 1367596963815</td>
<td>146</td>
</tr>
<tr>
<td>Receive : 1367596964988</td>
<td>Send: 1367596964858</td>
<td>130</td>
</tr>
<tr>
<td>Send: 1367596966032</td>
<td>Receive : 1367596966195</td>
<td>163</td>
</tr>
<tr>
<td>Receive : 1367596970310</td>
<td>Send: 1367596970140</td>
<td>170</td>
</tr>
<tr>
<td>Send: 1367596971219</td>
<td>Receive : 1367596971436</td>
<td>217</td>
</tr>
<tr>
<td>Receive : 1367596972541</td>
<td>Send: 1367596972379</td>
<td>162</td>
</tr>
<tr>
<td>Send: 1367596973505</td>
<td>Receive : 1367596973738</td>
<td>233</td>
</tr>
<tr>
<td>Receive : 1367596974760</td>
<td>Send: 1367596974610</td>
<td>150</td>
</tr>
<tr>
<td>Send: 1367596975892</td>
<td>Receive : 1367596976031</td>
<td>139</td>
</tr>
<tr>
<td>Receive : 1367596977501</td>
<td>Send: 1367596977273</td>
<td>228</td>
</tr>
<tr>
<td>Send: 1367596980361</td>
<td>Receive : 1367596980512</td>
<td>151</td>
</tr>
<tr>
<td>Receive : 1367596981704</td>
<td>Send: 1367596981485</td>
<td>219</td>
</tr>
<tr>
<td>Send: 1367596982950</td>
<td>Receive : 1367596983165</td>
<td>215</td>
</tr>
<tr>
<td>Receive : 1367596984157</td>
<td>Send: 1367596984008</td>
<td>149</td>
</tr>
<tr>
<td>Send: 1367596985140</td>
<td>Receive : 1367596985307</td>
<td>167</td>
</tr>
</tbody>
</table>

Average: 187.1 milliseconds
7.2.3 Final conclusion

<table>
<thead>
<tr>
<th>Test</th>
<th>Average (milliseconds)</th>
</tr>
</thead>
<tbody>
<tr>
<td>SQL nine games with 300 milliseconds</td>
<td>440.04</td>
</tr>
<tr>
<td>SQL one game with 300 milliseconds</td>
<td>447.59</td>
</tr>
<tr>
<td>SQL nine games with 0 milliseconds</td>
<td>421.19</td>
</tr>
<tr>
<td>SQL one game with 0 milliseconds</td>
<td>314.82</td>
</tr>
<tr>
<td>noSQL nine games with 300 milliseconds</td>
<td>368.3</td>
</tr>
<tr>
<td>noSQL one game with 300 milliseconds</td>
<td>341.7</td>
</tr>
<tr>
<td>noSQL nine games with 0 milliseconds</td>
<td>209.43</td>
</tr>
<tr>
<td>noSQL one game with 0 milliseconds</td>
<td>187.1</td>
</tr>
</tbody>
</table>

In this table is easy to see which option is better for do the connection with the DB (between redis cloud and ClearBD) in the game connect4: noSQL (redis cloud).

All the averages are better in the noSQL DB. The differences between one game or nine at the same time is only significative in SQL with polling of 0 milliseconds+response time, this can be because the number of person playing at the same time doing polling constantly difficult the connection with the server.

In the case of the noSQL, have nine games at the same time or only one is not significative but is important the time of polling, the differences between 300 and 0 milliseconds is about the double of time and this is a lot of time.

In conclusion, the best implementation for the connect four is noSQL (redis cloud) without polling (0 milliseconds).

- Unusual case: The average time in SQL nine games with 300 milliseconds is better than the SQL one game with 300 milliseconds, that is possible because the difference is very little and we have a small test.
8. Future work

To improve the games and get more people to play them we need to improve the interface and add some things. For example in the snake game it is a good idea add a section in the index to see the maximum score of the user friends. In the connect4 the graphics are only lines and circles, to improve this, we can change the board and the pieces for a good draws.

The next game to implement is a multiplayer game. In this case I talked about MMOG, this kind of game is a really challenge because normally this kind of game are very complex and needs to store a lot information. I talked about it in “6.4. MMOG (Massive Multiplayer Online Game)”. 
Appendix A: Privacy policy

Link: https://protected-wildwood-8266.herokuapp.com/privacy_Policy.html

<!DOCTYPE html>
<html lang="es">
<body>
<h1>Privacy policy</h1>
<p>This Privacy Policy was last revised on February 28th, 2013.</p>
<p>Privacy Policy explains what information of yours will be collected by html5 Snake and how the information will be used. I will not use or share your information with anyone except as described in this Privacy Policy.</p>
<p>The game is served on the Facebook Platform, so your activity on the game may sometimes be ruled by Facebook Privacy Policy and not ours. I, as developer, also respect Facebook Principle and Policies. You can read them on http://developers.facebook.com/policy/.</p>
<h2>Information I collect</h2>
<p>When you access the Game through Facebook, I may receive your UserID and the user IDs of the user's friends who have also connected with your application, I only need this for store your maximum score in the data base.</p>
<h2>How I use the collected data</h2>
<p>Just to save the maximum score.</p>
<h2>Additional Questions</h2>
<p>If you have any question, feel free to contact me at: roberto.martinez.q@gmail.com.</p>
</body>
</html>
Appendix B: Maintenance Manual

B.1: Directory Structure

B.1.1: Snake
→ Images: folder with the images needed for the application.
→ index.php: main page of the application.
→ javascript: folder with the javascripts.
  → game.js: main class of the game.
  → jquery-1.7.1.min.js: jquery library.
→ PHP: folder with the PHPs.
  → in.inc: file with the DB information.
  → regis.php: file which registers a new user using the user ID.
  → score.php: file to get the score of the user using his ID.
  → setScore.php: file to set the new score to the user with the ID received.
→ privacy_Policy.html: file with the information about my policy and privacy.
→ sdk: folder with the SDK of Facebook.
→ sound: folder with the sounds of the game.
→ styleSheets: folder with the css files for the application.
→ utils.php: file with functions to get some variables.

B.1.2: Connect4
→ Images: folder with the images needed for the application.
→ index.php: main page of the application.
→ indexnosql.php: main page of the application using the noSQL DB (redis cloud).
→ javascript: folder with the javascripts.
  → BoardCanvas.js: file which adds the listeners of the game.
  → GameBoard.js: file which draw the board and manage the board listeners.
  → GamePiece.js: file which draw the pieces.
GameState.js: file which manage the game state.
GameEngine.js: file which configure the game and contain the global variables.
WinnerDialog.js: file which draw the winner dialog.
jquery-1.7.1.min.js: jquery library.

PHP: folder with the PHPs.
in1.inc: file with the DB information.
getState.php: file to get the state of the game between the two users received.
getTurn.php: file to get the turn of the game between the two users received.
existGame.php: file to know if exists game between the two users received.
ewGame.php: file to create a game between the two users received.
saveSinc.php: file to increase the number of winning of the user.
saveState.php: file to save the new state in the game between the two users received.
predis-0.8: Folder with the Redis library to use Redis in PHP.
privacy_Policy.html: file with the information about my policy and privacy.
styleSheets: folder with the css files for the application.
utils.php: file with functions to get some variables.
nosql: this folder contain the images, javascripts and PHPs with the noSQL DB.

B.2 Make it work
To make it work we just need to upload it to a host and run it using the host domain. This host should provide a SSL because Facebook require it. The best way to upload this application is go to the Facebook developers page, create a new application using Heroku and upload the application in Heroku.
References/Bibliography

[1] x10hosting. [online] Available at: https://x10hosting.com/ [Accessed 15 April 2013]


[21] [online] Available at: https://www.facebook.com [Accessed 25 April 2013]

[22] [online] Available at: https://devcenter.heroku.com/articles/facebook [Accessed 25 April 2013]


[28] w3Schools. [online] Available at: http://www.w3schools.com/default.asp [Accessed January 2013]
## Glossary

<table>
<thead>
<tr>
<th>Term</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>SSL</td>
<td>Secure Sockets Layer, that provide communication security over the internet.</td>
</tr>
<tr>
<td>MMOG</td>
<td>Massive Multiplayer Online Game. The online games who a lot of people plays at the same time are MMOG.</td>
</tr>
<tr>
<td>Script</td>
<td>Command file.</td>
</tr>
<tr>
<td>DB</td>
<td>Data base. This store the information.</td>
</tr>
<tr>
<td>Hash</td>
<td>A map of string keys and string values.</td>
</tr>
<tr>
<td>app</td>
<td>Application.</td>
</tr>
</tbody>
</table>