

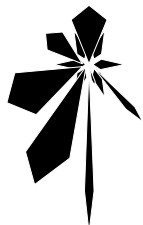
# Introduction to Browser- Based Online Game Design and Development

Revision 1.1

June 22, 2010

by

Aloysius Indrayanto



(C) 2010 AnemoneSoft.com

This document is multi-licensed under the Creative Commons Attribution Share-Alike (CC-BY-SA) license version 3.0 and the GNU Free Documentation License (GNU FDL) version 1.3 or later.

## 1. Introduction

A game is an activity done by a group of peoples mainly for enjoyment and satisfaction. In a game, the peoples involved are called players. Each player (or a group of players) make decisions based on some predefined rules in order to achieve some predefined objectives. A game is a structured play. Therefore, sometimes, games can be used as educational tools. A game that is played using a computerized system is usually called a video game.

An online game is a video game which is played by multiple players across a computer network. In a simple and private online game, the network can be formed by joining several computers together to form a Local Area Network (LAN). In a public online game, the network is usually the internet. In this configuration, there will be a public server (or a cluster of servers) to where players can connect to and play the game.

## 2. Classification of Online Game

There are various methods to classify online games. Two of the methods are classification based on the platform and classification based on the genre.

### 2.1. Classification of Online Game Based on Their Platform

Online games can be classified based on their platform (where they can be played). This classification is basically done by defining the applications or tools that will be needed to play the games. Hence, using this method, an online game can be classified as:

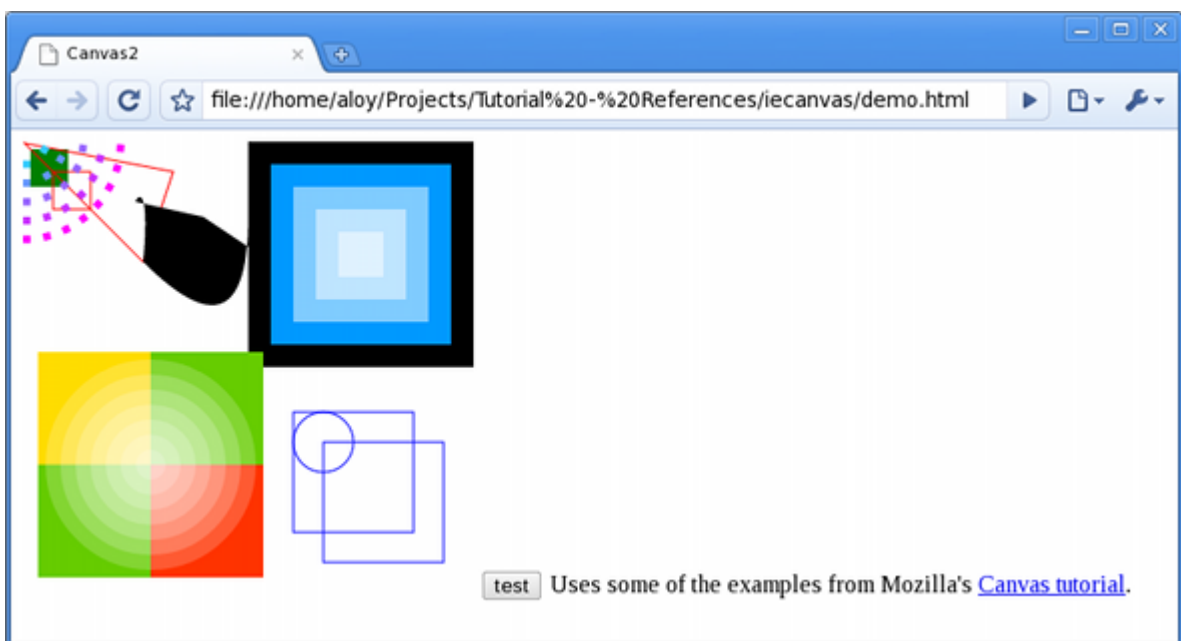
1. **A game which is played using an install-able client application.** This type of game usually generates a quite complicated sequence of animations. Virtually all true/full 3D online games are usually of this type.
2. **A game which is played using a web browser.** This type of game can be further divided into two types:
  - a. A game that requires a browser plugin.
    - The most famous plugin for browser-based online game is the Flash player. It can be used to create (play) both 2D and semi 3D (2.5D) games. True/full 3D game is also theoretically possible. However, with the currently available Flash players, the performance would not be so good.

- The second plugin in the Java applet plugin. Basically, due to Java can access the system graphics card's capability (with Java OpenGL), it is possible to create a true/full 3D game using Java applet. However, Java is a quite complex programming language to learn.
- b. A game that does not require a browser plugin. This type of game is usually implemented fully using HTML and JavaScript. Basically, it is possible to create a good 2D (or 2.5D) game using JavaScript.
  - With the new HTML 5 <canvas> tag, creating a 2D game with the same picture quality with a Flash game is now possible. Currently, <canvas> tag only provides 2D rendering context. Therefore, 2.5D and 3D games can be created by using software renderer implemented on top of the 2D rendering context. However, in the future, a native 3D rendering context may be also provided using WebGL. Hence, it is not impossible that in the future a real 3D Role Playing Game (RPG) can be played through web browser.
    - Firefox 3.0+, Google Chrome 3.0+, Apple Safari 3.2+, and Opera 9.6+ should already provide 2D rendering context with all the W3C-standardized basic canvas API.
    - Firefox 3.5+, Google Chrome 4.0+, Apple Safari 4.0+, and Opera 10.5+ should already provide 2D rendering context with the W3C-standardized text rendering API.
    - Internet Explorer up to IE 8 has not yet support the <canvas> tag. However, it is possible to create an emulation layer for many of the canvas' basic functionality using IE's support for Vector Markup Language (VML) and JavaScript. Please refer to the URL in the reference section ([explorercanvas](#) and [excanvas-patch](#)) for more details about this matter.
  - With the new HTML 5 <audio> tag, playing audio natively in JavaScript-based game is now possible.
    - Firefox 3.5+, Google Chrome 3.0+, Apple Safari 4.0+, and Opera 10.5+ should already provide full support for the <audio> tag.

- Internet Explorer up to IE 8 has not yet support the <audio> tag. However, it is possible to create a JavaScript library that will fallback to the <embed> tag to play audio in Internet Explorer.
  - With the new HTML 5 <video> tag, playing video natively in JavaScript-based game is now possible.
    - Firefox 3.5+, Google Chrome 3.0+, Apple Safari 4.0+, and Opera 10.5+ should already provide full support for the <video> tag.
    - Internet Explorer up to IE 8 has not yet support the <video> tag. However, it is possible to create a JavaScript library that will fallback to the <embed> tag to play video in Internet Explorer.
3. **A game that is played using e-mail.** This type of game can be played by receiving the game's event messages and sending the response messages by e-mail. Basically, this type of game will be a simple, text-based game. Hence, this type of game would have a very limited number of users. In addition, this type of game is not a true online game because of the ability to store e-mails for later sending or retrieval.

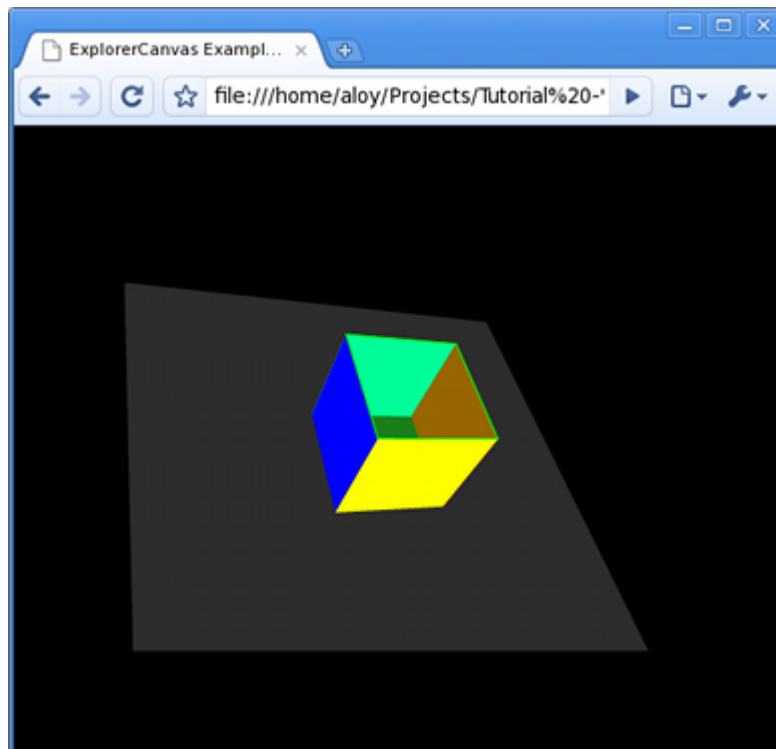
### 2.1.1. Examples of HTML 5's Canvas Capabilities

The screenshot below shows the capability of HTML 5's canvas to draw points, lines, rectangles, polygons, and circles using various colors and transparency.



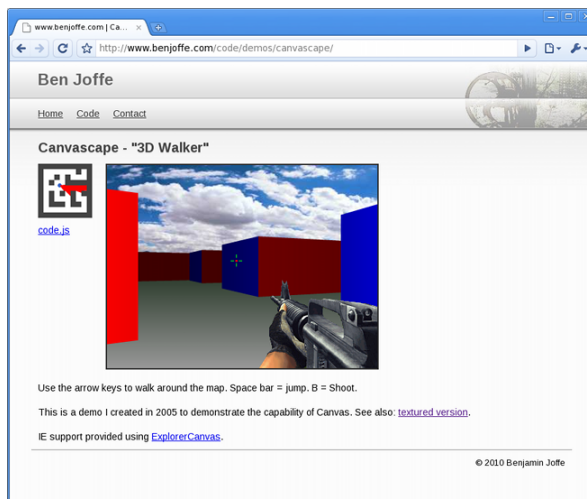
<http://me.eae.net/stuff/iecanvas>

The screenshot below shows the capability of HTML 5's canvas to render a simple, rotating 3D box using a simple software renderer.

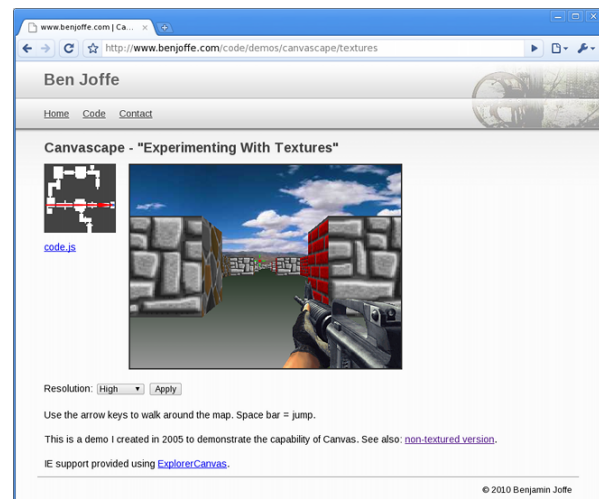


<http://code.google.com/p/explorercanvas>

The screenshots below show the capability of HTML 5's canvas to generate a simple, walk-able 3D scene that looks like a first-person shooter game. The scene can be rendered with and without texture using a simple software renderer.



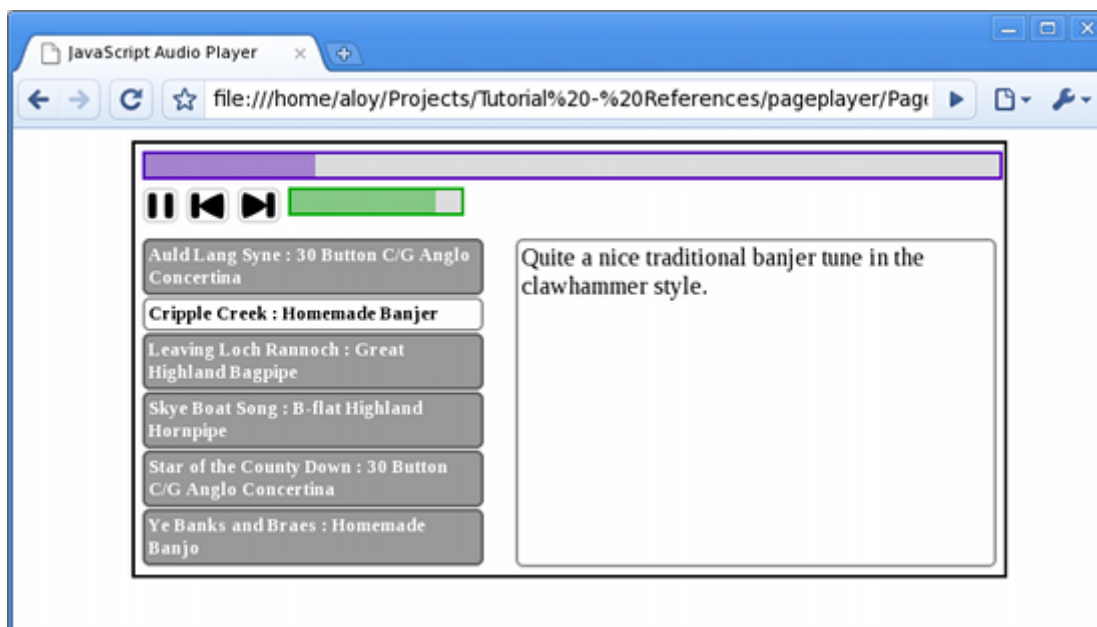
<http://www.benjoffe.com/code/demos/canvascape>



<http://www.benjoffe.com/code/demos/canvascape/textures>

### 2.1.2. Examples of HTML 5's Native Audio and Video

The screenshot below shows an audio player implemented entirely using HTML 5 and JavaScript.



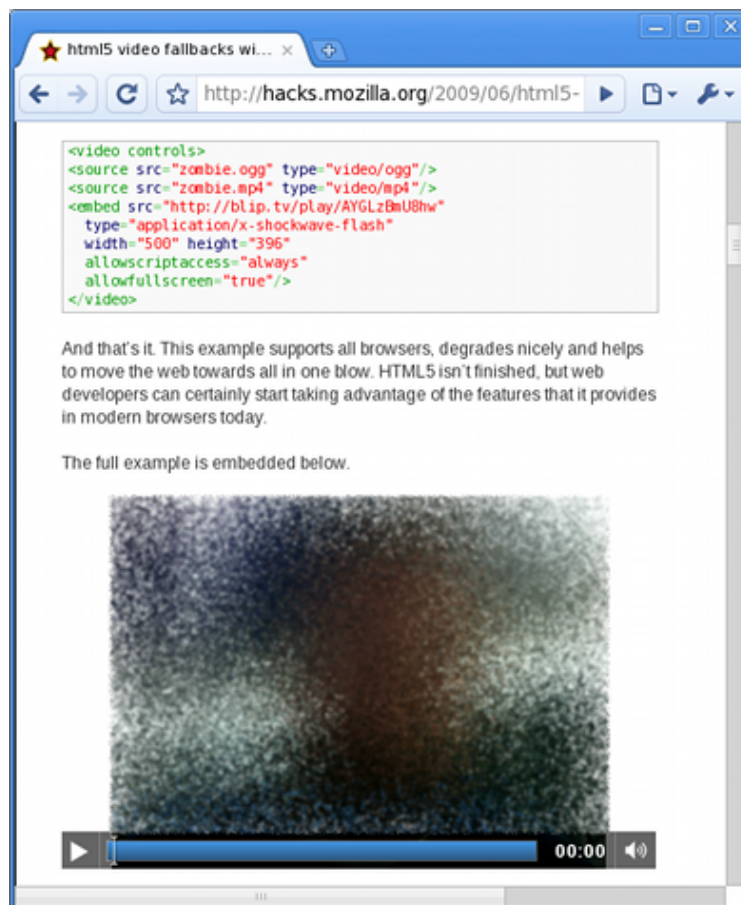
<http://www.jezra.net/projects/pageplayer>

Currently, only very few audio codecs that are natively supported by browsers which support the HTML 5's `<audio>` tag (other codecs would require additional installation). The table below lists the audio codecs that are natively supported by those browsers.

Browser	Ogg Vorbis	MP3	WAV
FireFox 3.5	✓		✓
Google Chrome 3.0	✓	✓	
Apple Safari 4.0		✓	✓
Opera 10.5			✓

Internet Explorer does not supports the `<audio>` tag, however audio can be played using the `<embed>` tag that is generated dynamically using JavaScript. The list of codecs supported by IE will depend on the installed codecs in the system.

The screenshot below shows a video player implemented entirely using HTML 5 and JavaScript. Note that the rendered video is intentionally blurred in this tutorial because it contains somebody's face.



<http://hacks.mozilla.org/2009/06/html5-video-fallbacks-markup>

Currently, only very few video codecs that are natively supported by browsers which support the HTML 5's <video> tag (other codecs would require additional installation). The table below lists the video codecs that are natively supported by those browsers.

Browser	Ogg Theora	H.264
FireFox 3.5	✓	
Google Chrome 3.0	✓	✓
Apple Safari 4.0		✓
Opera 10.5	✓	

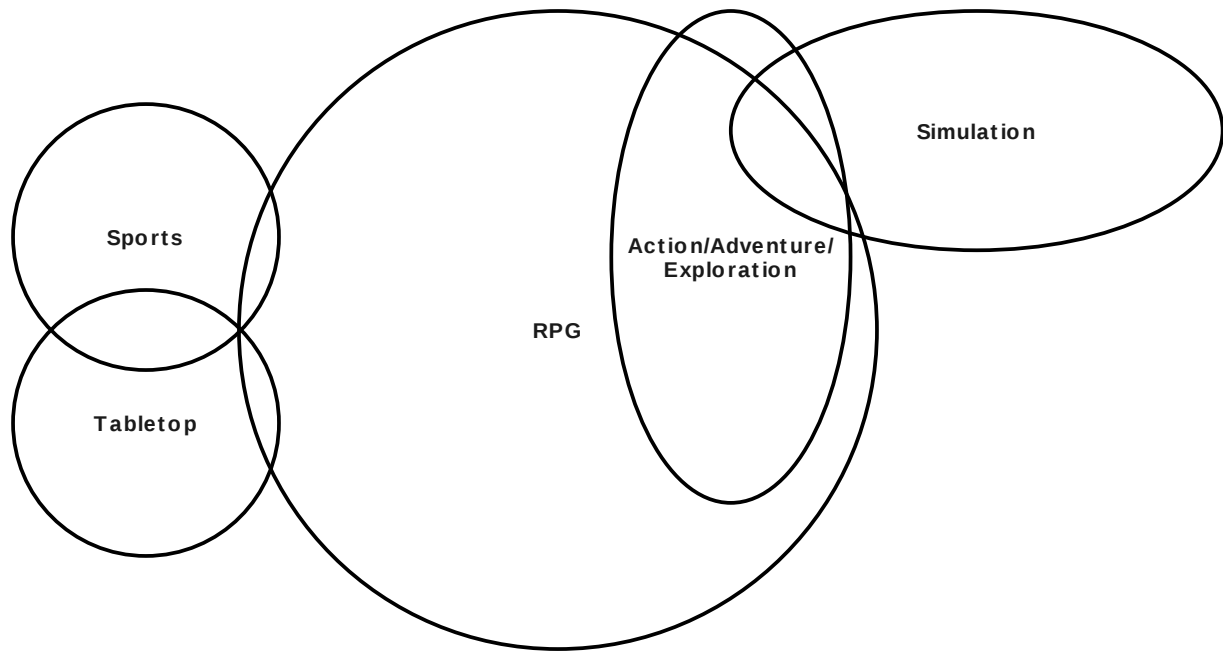
Internet Explorer does not support the <video> tag, however video can be played using the <embed> tag. The list of codecs supported by IE will depend on the installed codecs in the system.

## 2.2. Classification of Online Game Based on Their Genres

Online games can be classified based on their genres. Hence, using this method, an online game can be classified as:

1. **A Role Playing Game (RPG).** This may be the most complicated type of online game. In this game, the player take a role as a virtual person in the game world. The story element and character-presence in an RPG game are strong, as if, the played character does life in the virtual world. The player must adhere to the rules defined in the game world (just as we all bound to physics laws as well as our country's rules and regulation). Action and adventure are usually part of an RPG game. Puzzles and mini-game can be also included in the gameplay. Basically, using the given resources and rules, the player will need to develop his character according to a predefined set of goals.
2. **Action, adventure, and exploration game.** This type of game is more simpler than an RPG game. Basically, the key element in this game is the action/adventure/exploration. A bit of story can be embedded in the gameplay, but not as strong as the story embedded in an RPG game.
3. **Simulation game.** This type of game can include a broad area, from flight simulation to virtual farm and restaurant. Basically, this type of game can have some thrill and action such as in flight simulator, or, can also offer a more calm and fun environment such as in virtual farm/restaurant/etc. A much more complicated form of this game, such as business simulation, virtual life, etc. can also be implemented.
4. **Sports game.** This type of game emphasize on the enjoyment of virtual sports. The topic in this game can range from football, tennis, golf, etc. to the other form of sports such as chess, cards, etc. However, the latter type of sports can also be categorized as tabletop game.
5. **Tabletop game.** This type of game can include chess, cards, or even pinball. Basically, this type of game can be played more seriously (for example, chess), or, just for fun (for example, pinball).

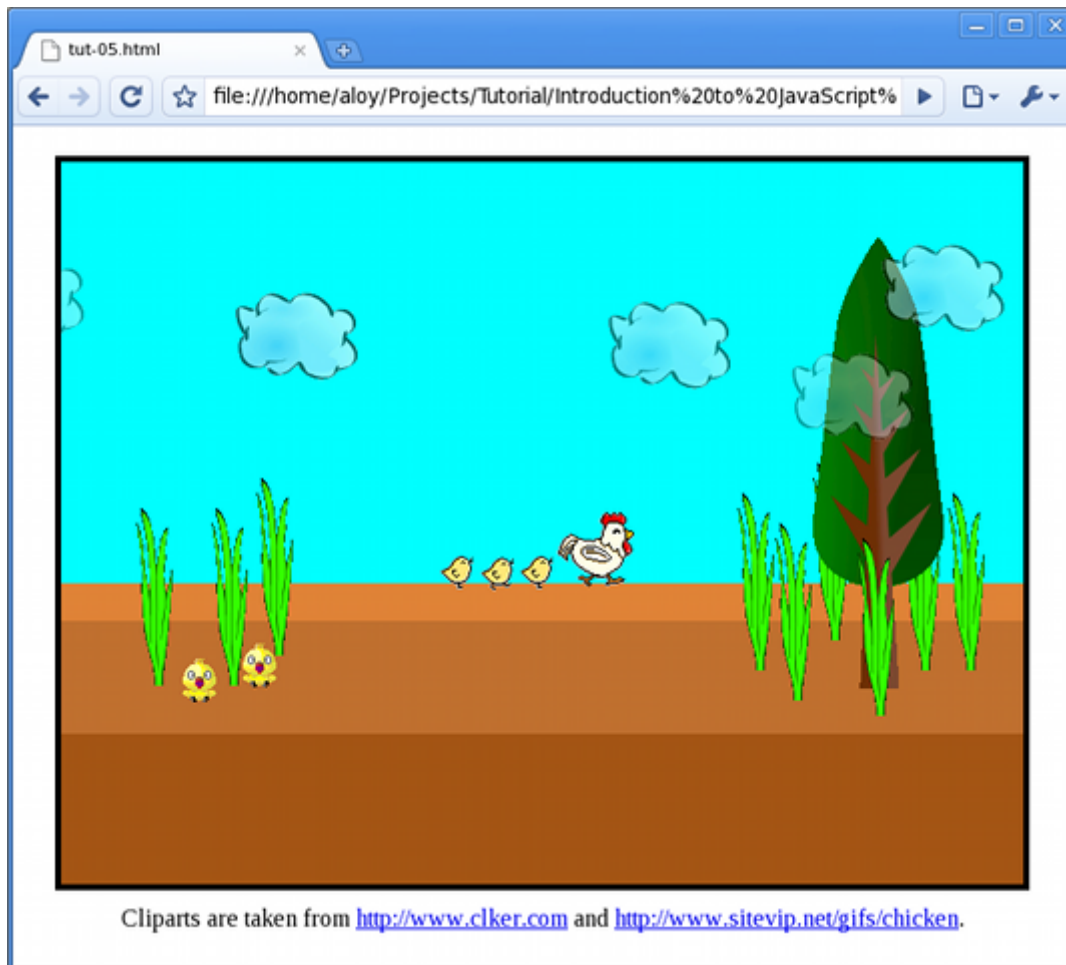
The picture below shows a possible interaction between the above genre of games. Basically, an RPG game usually includes action/adventure/exploration. An RPG game may also includes sports and tabletop games as mini-games as well as some form of simulation. Sports and tabletop game form an intersection (for example, chess is both sport and played on top of a table). Finally, a simulation game may also include some form of action/adventure/exploration.



With the current available browser technology, it would not be feasible to implement 3D sports games that need rapid and detailed movements such as football, tennis, etc. However, it is possible to implement all the other type of games to some degree.

Implementing an RPG game with just HTML 4.01 JavaScript can be a very big challenge. However with the adoption of HTML 5, it would not be that difficult to implement a true RPG game entirely in web browser. In the future with the adoption of WebGL, even 3D sports games that need rapid and detailed movements can be implemented in web browser.

Implementing a 2D game using only HTML 4.01 and JavaScript is pretty straightforward. The screenshot below shows an animated scenery from our other tutorial (Introduction to JavaScript Animation). Basically, by using animated GIF for images and controlling the images position using JavaScript it is possible to create a good 2D animation. In order to convert this animation into a game, what is needed is basically to add the capability to capture user inputs and transform them into “some change in animation” using JavaScript. Of course some rules (and may be a simple artificial intelligence) would also need to be embedded in the application.



## 3. Beginning a Browser-Based Online Game Development

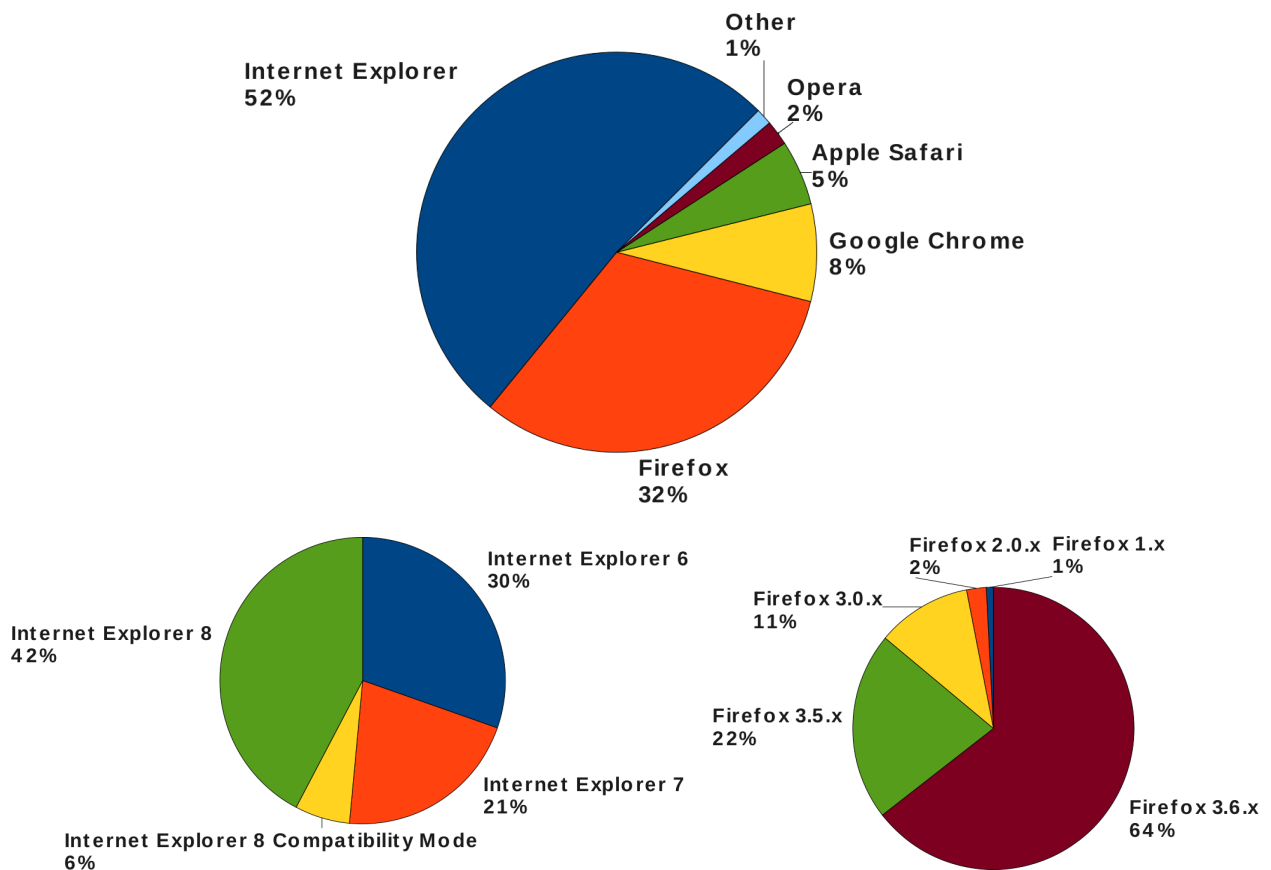
### 3.1. Choice of Platform

Developing an online game that with an install-able client application usually requires more efforts than developing a browser-based online game. This is due to the fact that the client application is usually developed using C++ which is a much more complicated language to learn than JavaScript. For 3D online game, the requirement will increase. A developer will also need to be familiar with 3D application development toolkit such as OpenGL as well as some 3D mathematics.

In this tutorial, we will discuss about the design and development of browser-based online games that requires only JavaScript. Basically, the choice of HTML (version 4.01 versus version 5) and JavaScript features would depend on the genre of the game. Developing a pure 2D game can be done by just using HTML 4.01 and JavaScript. However, developing a 2.5D and 3D (using software renderer) games would need to use the features of HTML 5 (canvas). Also, due

to the slight incompatibilities in JavaScript implementation between different browsers, it would be a good idea to use a JavaScript library/framework that can provide a more uniform API across different browsers.

The picture below show the usage share of web major browsers per May 2010 according to Wikipedia.



Using the above pictures as references, it would be good to develop a browser-based online game that can run under:

- Firefox version 3.x and above.
- Internet Explorer version 6 and above (IE version 5.5 and lower are virtually extinct).
- Google Chrome version 3.0 and above.
- Apple Safari version 3.2 and above.
- Opera version 9.6 and above.

However, if HTML 5 features (canvas and native audio/video) are needed, then it would be good enough to develop a browser-based online game that can run under:

- Firefox version 3.5 and above.

- Google Chrome version 4.0 and above.
- Apple Safari version 4.0 and above.
- Opera version 10.5 and above.

Support for canvas under Internet Explorer can be provided using an emulation layer. Please refer to the URL in the reference section ([explorercanvas](#) and [excanvas-patch](#)) for more details about the canvas emulation layer. Support for audio and video in Internet Explorer still can be routed to the <embed> tag using JavaScript.

### **3.2. Choice of Genre and Topic**

Some online games may only need simple graphics, while some others may need very complicated graphics. An RPG game would usually need the most complicated graphics. Depends on the skill of the developers, choosing the right genre will be important. A developer with a good character-drawing skill may choose to create an RPG game from the start; on the contrary, a developer with an average character-drawing skill should choose a simpler genre.

Despite the chosen genre, it is very important to find a topic (story or gameflow) that are interesting to both the developers and target user. Basically, browsing existing online games of the same genre would give an idea of what topics currently exist and played by players around the world. Note that, it would not be a good idea to just imitate or combine existing topics/features. Adding variation and special characteristic of your own are very important.

### **3.3. Life Cycle of a Browser-Based Online Game Product**

There is nothing fixed about the life cycle of a product. However, basically the life cycle of a browser-based online game can be described as:

1. **Internal development stage.** In this stage, the game is not yet available for general public. It is possible to release (publish in web) the alpha version of the game for private/limited number of testers. However, making any alpha release public is generally a bad idea.
2. **Beta release.** There should be no more “show-stopper” (fatal) bugs for the beta release of the game. The game should also already feature complete (the game must be completely playable from start until finish). Basically, it can be a good idea to make all/some of the beta releases available for general public. Do not forget to start advertising the game.

Provides an e-mail address on the the game web pages so that users can give feedback and report bugs.

3. **Minor improvement and bug fixes.** No complicated/big features that may cause instability shall be added after the first beta release. Adding minor features to make the gameplay better is good as long as those features do not change the core flow/working/logic of the game. Bug fixing to remove the remaining annoying (but not fatal) bugs should be done now.
4. **Release candidate.** Basically, all game features should be “frozen” in this release. There should be no more feature addition after this release, except if that feature addition is the part of a bug fixing process. Doing more advertisement for the game is a good idea.
5. **Bug fixes.** Fix all remaining visible bugs to prepare for the version 1.0.0 release.
6. **Version 1.0.0 release.** While there is no application that completely bugs free, in this release we need to ensure that there is no more user visible bug in the game application. If we happen to find such a bug, fix it immediately and update the game. Doing some more advertisement for the game is a good idea.
7. **Maintenance stage.** This stage is basically the “run time” stage of the game. The possible actions in this stage will be iterated over and over. Some actions that can be done in this stage:
  - **Content addition.** Basically, an online game needs to be designed so that contents can be added easily without changing the program logic. Adding contents regularly would be a good method to attract users. Do not forget to advertise the newly added content.
  - **Minor improvement.** Adding small improvements such as better user interface, better images and sound effects, etc. would be a good idea as long as they do not change the core program logic. It would be a good idea to add the improvements step by step (not all at once). If all the improvements are all added at once, in the future, it would be much more difficult to find a new idea for improvement.
  - **Noticeable improvement.** Noticeable does not mean major. It means that the added features will looks big and attract users. An example of this improvement is a user interface overhaul that will be directly noticeable by all players. A user interface overhaul does not necessarily modify many part of the program logic, they mostly can be done by providing a much better looks for buttons, icons, etc.

8. **Declining stage.** Basically, the end of life of an online game can be quite long. This is due to the fact that when old players no longer play the game, there can be new players registering for the game. Some actions can be done in this game to further the life time of the game:

- **More advertisement.** This can be done to attract new players.
- **Content recycle with discount.** Older content (especially payed contents) can be re-included/exhibited so that newer players can have chances to buy them. If appropriate, the price of payed contents can be discounted.
- **Major feature addition.** This would depend on the genres and topics of the games. For example:
  - In a space exploration/conquest game: adding a new galaxy to explore/conquer, adding new races/enemies, etc.
  - In a virtual farm game: adding a new type of farm that have a totally different looks and characteristics with the original farm.

Basically adding this major features would usually requires changes in the core program logic and introduces new bugs. Therefore, do not forget to inform the users before actually updating the game. Having a separated version number for the new features is also possible (for example: the new features will be released in beta version while the original game is kept in its original version).

### 3.4. Revenue Model

Nowadays, most (if not all) browser-based online games can be played freely. Revenue can be obtained by making few parts of the game non free. However, the game itself must be completely playable without major annoyance even if a player has decided to be a free player forever. A non-free player will of course get some privileges.

Basically, there are two major revenue models for browser-based online game that can be used by game developers; they are:

1. **Subscription.** This is the older revenue model. Basically, a player can subscribe for an additional features/privileges in the game. The features/privileges can include:
  - The ability to buy special equipment/items/etc. that have much higher status/power;
  - The ability to use certain services such as recharging energy, health, etc.

2. **Pay per item/service.** This is the newer revenue model. Basically the additional features/privileges offered is the same with the subscription model, however instead of regular renewing their membership, players will pay for each item/service they buy/use one by one. Of course, giving some discounts if a player decided to buy more than one items at once can be a good idea.

So, how a player can buy for special items/services? In the case of subscription model, players can simply renew their membership by paying a certain amount of money. The payment can be done by utilizing online service such as PayPal and Google Checkout or by direct transfer to the game developer's bank account.

In the case of pay per item/service, the existence of virtual currency will be a good idea. Basically, a player buy the virtual currency using real money and then use the virtual currency to buy the special items/services. Buying virtual currency can be done by two ways:

- By utilizing online service such as PayPal and Google Checkout. In this case, the game system will need to be able to handle transactions automatically. After PayPal or Google Checkout verify the validity of the payment, they can be asked to inform our game server. Hence, the game server can update the amount of virtual currency owned by the particular user. There are many other online payment services, however, ensure that they can be trusted before you decided to use them.
- By using game coupon/voucher. In this case, a player can buy game coupons/vouchers to refill his/her virtual currency.

### **3.5. Needed Tools**

A list of tools that would be needed for developing a browser-based online game is shown below. Note that, depend on the game, may not all tools will be needed.

- A local development machine with an installation of Apache Web Server (HTTPD) version 2.2.x, PHP: Hypertext Preprocessor version 5.x, and MySQL Community Server version 5.x. Other web server, server-side scripting language, and database vendor can also be used. However, the combination of Linux, Apache, MySQL, and PHP (LAMP) is among the most popular solutions for web/application server (in Windows, it is called WAMP).
- A recent enough browser such as Firefox 3.5+, Google Chrome 4.0+, Apple Safari 4.0+, Opera 10.5+, and Internet Explorer 7.0+.

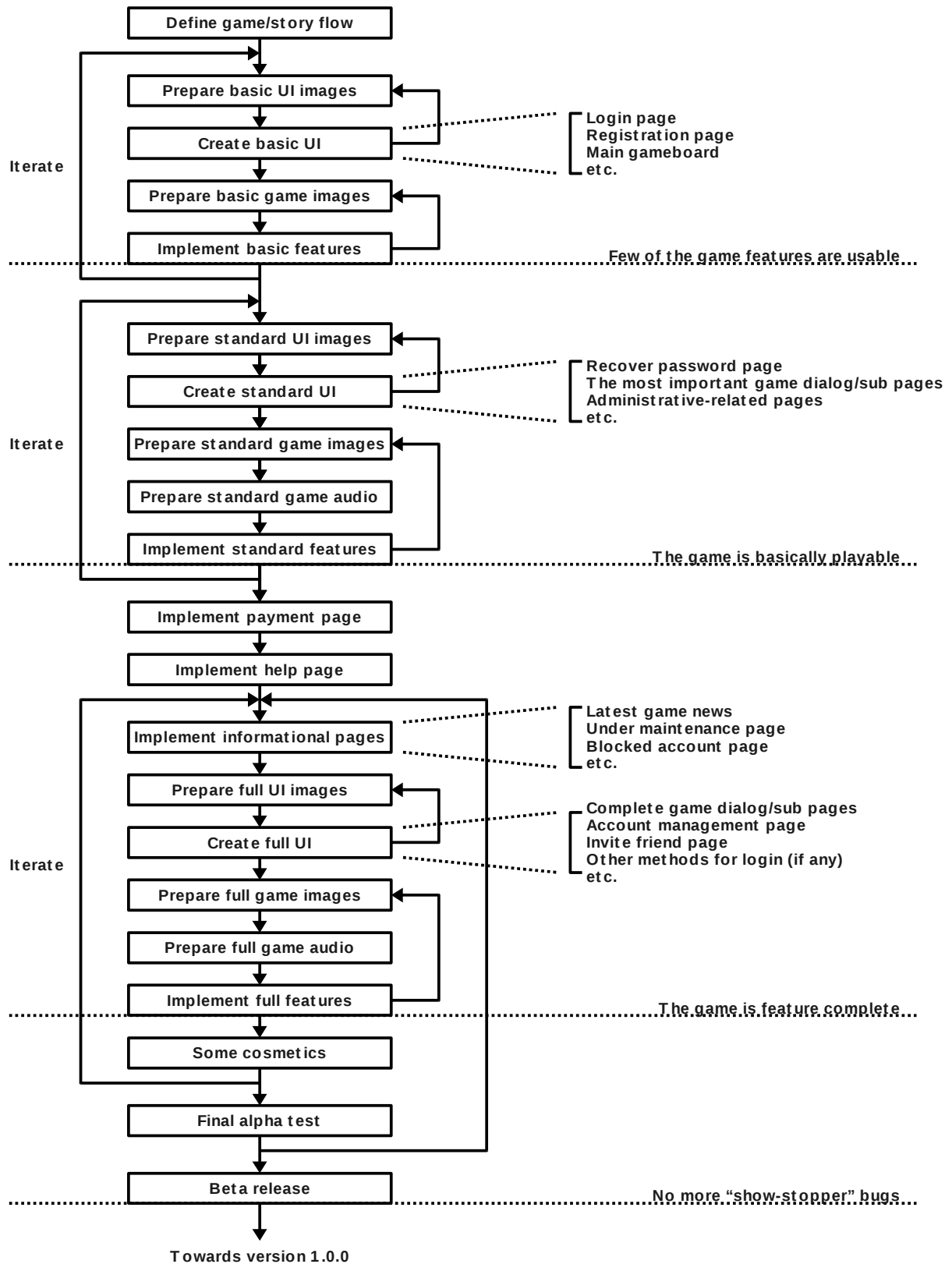
- A JavaScript compatibility framework/library such as Prototype.JS and `explorercanvas` with `excanvas-patch` (please refer to the URL in the reference section).
- A 2D graphics authoring tool such as the GNU Image Manipulation Program (<http://www.gimp.org>) for developing button images, icons, character pictures, items, etc. for the game.
- A 3D model authoring tool such as the Blender 3D Content Creation Suite (<http://www.blender.org>) for developing the character models, items, etc. for the game.
- An audio editing tool such as Audacity (<http://audacity.sourceforge.net>) for editing sound effects and converting audio file formats (if the game needs audio).
- A MIDI authoring software such as Rosegarden (<http://www.rosegardenmusic.com>) if you would like to compose your own background music.
- Text/code editor with syntax highlighting for HTML, CSS, JavaScript, and PHP.

### 3.6. Needed Skills

A list of skills that would be needed for developing a browser-based online game is shown in the list below.

- **Imagination.** There is no game without imagination. Indeed, there are games with real-life topics, however their market might be limited.
- **Scenario/flow design.** It is important to be able design how the game will be played. This includes the main story (if needed by the genre) and the progression-flow of the player in the game. Interconnection/flow between the game's user interface (dialog) will also need to be designed.
- **Programming.** Client-side (JavaScript) and server-side (PHP) programming skills will be needed. There is no need to be an advance programmer on those language because programming skills will improve as time passes. However, continuous learning, especially using materials in the internet is very important.
- **Graphics and audio authoring.** Having a skill to create 2D/3D models as well as sound/audio effect is a good idea. Due to music and sound effect authoring is more difficult than 2D/3D model authoring, using a royalty free music and sound effect in the internet will be a good start. Just remember to give credits to the original authors.
- **A bit of management.** There is no need to have advance management skill. Knowing how to manage oneself and to manage a small software project is good enough.

### 4. Flow of a Browser-Based Online Game Development



Basically there would be four major milestones the start of the game development until the first beta release. They are:

1. The game reach a preview stage where few of the game's features are usable.
2. The game reach an intermediate stage where the game has basically become playable (all the important game web page/dialog/etc. needed to play the game are implemented).
3. The game reach a final development stage where it becomes features complete.
4. The game reach a beta release stage where there shall be no more “show-stopper” (fatal) bugs.

It is possible to achieve those four milestones in about two to three months for simpler genre (action/adventure/exploration, simulation, and tabletop). However, expect that the needed time would double/triple for an RPG game.

After the beta stage, there should be no major features added to the game. After the release candidate, the game feature is “frozen”; only bug fixes are allowed. Finally, after the release of version 1.0.0, the game will be in maintenance mode where only content additions and minor improvements that do not change the core game logic are allowed.

The picture in the previous page shows a possible flow chart of game development. Each milestone is marked with a long, dotted, horizontal line. Iterations between stages is common. Basically, iterations are needed to ensure that the game is really ready to reach a particular milestone. Final beautification (cosmetics) for the game interface can be done before the final alpha test. It is necessary to test the game again after the beautification because this process may cause some new or hidden bugs to appear. The first beta release can be done after the game passes the final alpha test.

After the beta release, do not forget to start advertising the game. Provides an e-mail address on the game's web pages so that users can give feedback or report bugs. If you are unsure of how to advertise the game, start by telling your friends about the game and ask for their help to test the game and gives some feedback. Posting news about the game in your blog or in a social network site are also a good idea. It is possible to also use payed advertisement after the beta release, however, unless you are sure that the games no longer has annoying bugs, it may be better to delay using payed advertisement until the version 1.0.0 release.

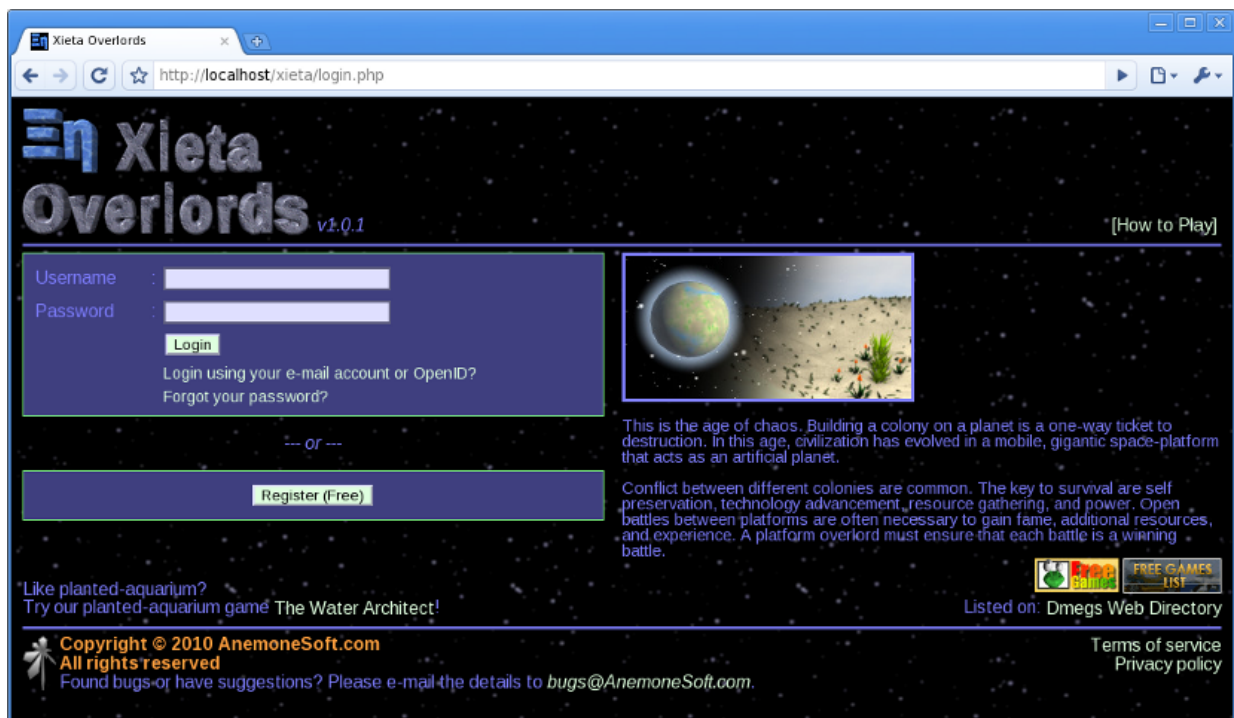
## 5. Some Considerations

There are some considerations of how an online game should be developed/presented. This consideration is not fixed rules, just a bit of personal opinion and experience:

1. Consider to use existing framework (for both JavaScript and PHP) if possible. If you need customized libraries, writing them yourself is OK. Just ensure that they do not take too much time and effort.
2. It is a good idea to put your company logo, name, and e-mail (for suggestion and bug reporting) in all of the game web pages. However, please ensure that those images and text will not annoy the user. This can be done by placing these contents at the bottom part of the game web page, together with the game's "term of services", "privacy policies", "payment terms", etc. Also, ensure that the company logo is not too large as it may obstruct the user.
3. It is a good idea to put the game logo and title at the top of each web page. For the index/login page, the logo and title can be made larger (occupy about 25% of the monitor's height). However, for all other pages, they must be made smaller (occupy only about 10% of the monitor's height).
4. Maximize the "real" gameplay area. This means that if your game will need to display status and/or menu at the top of the gameplay area, make sure that they occupy as little space of possible. Basically, together with the game logo and title, the status and/or menu shall be made not to occupy more than 25% of the monitor's height.
5. Advertisement in the game is allowed. In the index/login page, advertisement banners can be placed on the top of the page (side to side with your game's logo and title). However, advertisement inside the gameplay-related pages must not obstruct the player. Hence, it is a good idea to put the advertisement banner at the bottom-most part of the page, after the company logo, name, contact e-mail, and the game's "term of services", "privacy policies", "payment terms", etc.
6. What should be in the index/login page? Basically these item/features shall be put in the page:
  - The login form (of course);
  - A link from where users can recover their passwords;
  - A link from where new users can register;

- Some background story about the game;
- A link to your company's main web site and e-mail (if any)';
- A link to the game's help page;
- Links to the game's “term of services”, “privacy policies”, “payment terms”, etc.
- Links to your other game (if any, optional);
- Advertisement banners (optional, do not put too many of them);

The picture below shows a possible example of the login page of a browser-based online game.



## References

- <http://en.wikipedia.org/wiki/Game>, June 19, 2010
- [http://en.wikipedia.org/wiki/Online\\_game](http://en.wikipedia.org/wiki/Online_game), June 15, 2010
- [http://en.wikipedia.org/wiki/Flash\\_player](http://en.wikipedia.org/wiki/Flash_player), June 15, 2010
- [http://en.wikipedia.org/wiki/Java\\_\(programming\\_language\)](http://en.wikipedia.org/wiki/Java_(programming_language)), June 19, 2010
- [http://en.wikipedia.org/wiki/Java\\_applet](http://en.wikipedia.org/wiki/Java_applet), June 19, 2010
- [http://en.wikipedia.org/wiki/Java\\_OpenGL](http://en.wikipedia.org/wiki/Java_OpenGL), June 20, 2010
- <http://www.khronos.org/webgl>, June 20, 2010
- <http://en.wikipedia.org/wiki/HTML5>, June 16, 2010
- <http://www.sergiopereira.com/articles/prototype.js.html>, June 01, 2010
- [http://www.deepbluesky.com/blog/-/browser-support-for-css3-and-html5\\_72](http://www.deepbluesky.com/blog/-/browser-support-for-css3-and-html5_72), June 16, 2010
- <http://caniuse.com>, June 20, 2010
- <http://html5doctor.com/native-audio-in-the-browser>, June 20, 2010
- <http://www.jezra.net/projects/pageplayer>, June 20, 2010
- <http://me.eae.net/stuff/iecanvas>, June 16, 2010
- <http://code.google.com/p/explorercanvas>, June 16, 2010
- <http://www.sencha.com/playpen/tm/excanvas-patch>, June 20, 2010
- <http://www.benjoffe.com/code/demos/canvascape>, June 16, 2010
- <http://hacks.mozilla.org/2009/06/html5-video-fallbacks-markup>, June 20, 2010
- [http://en.wikipedia.org/wiki/Vector\\_Markup\\_Language](http://en.wikipedia.org/wiki/Vector_Markup_Language), June 20, 2010
- [http://en.wikipedia.org/wiki/Usage\\_share\\_of\\_web\\_browsers](http://en.wikipedia.org/wiki/Usage_share_of_web_browsers), June 20, 2010
- <http://en.wikipedia.org/wiki/Template:Msieshare1>, June 20, 2010
- [http://en.wikipedia.org/wiki/Template:Firefox\\_usage\\_share](http://en.wikipedia.org/wiki/Template:Firefox_usage_share), June 20, 2010
- [http://en.wikipedia.org/wiki/Software\\_release\\_life\\_cycle](http://en.wikipedia.org/wiki/Software_release_life_cycle), June 20, 2010

<http://en.wikipedia.org/wiki/PayPal>, June 20, 2010

[http://en.wikipedia.org/wiki/Google\\_Checkout](http://en.wikipedia.org/wiki/Google_Checkout), June 20, 2010

<http://httpd.apache.org>, June 09, 2010

<http://php.net/index.php>, June 09, 2010

<http://www.mysql.com>, June 09, 2010

<http://www.wampserver.com/en>, June 09, 2010

[http://en.wikipedia.org/wiki/LAMP\\_\(software\\_bundle\)](http://en.wikipedia.org/wiki/LAMP_(software_bundle)), June 09, 2010

<http://en.wikipedia.org/wiki/WAMP>, June 09, 2010