Institutionen för datavetenskap
Department of Computer science

Examensarbete

Performance test and optimize in HTML5-based web game: a case study of Flappy Bird

by

Ping Liu

LIU-IDA/LITH-EX-A--15/001--SE

2015-11-18

Linköpings universitet
SE-581 83 Linköping, Sweden

Linköpings universitet
581 83 Linköping

Linköpings universitet

Linköpings universitet
Institutionen för datavetenskap

Examensarbete

Performance test and optimize in HTML5-based web game: a case study of Flappy Bird

by

Ping Liu

LIU-IDA/LITH-EX-A--15/001--SE

2015-11-18

Handledare: Min handledare
Examinator: Min examinator

Abstract

Abstract.tex

Acknowledgments

Acknowledgments.tex

Contents

List of Figures

List of Tables

1 Introduction

Since the HTML5 released and especially new element called Canvas, The percentage of web games in the whole video gaming market is increasing day by day. And what's more, the hot market return to the improvement of web game developing environment, such as open source web game engine and some useful web libraries. With Canvas and WebGL, the visual effect of games on website can be as good as that on the desktop application.

This thesis will take Flappy Bird as a case study, Flappy Bird was released in May 2013 and in early 2014 it got a amazing rise in popularity and became the most downloaded free game in the IOS App Store. Now, there are different versions realized on website and we will take hyspace's flappy as a case study. The reasons for taking this game are fellows: firstly, it is open source you can see the code and you can take white box test, secondly, this game itself is quite simple but the code itself is quite typical of misunderstanding and wrong use of graphics function.

1.1 Motivation

Although, a lot of plugins and open source game engine can help us build a simple game very fast, but with the misunderstanding of Canvas and WebGL and also the wrong use of these plugins, and actually nowadays in order to attract customers and developing interesting games, the size of application inscreases rapidly, these application hit the performance wall. It is easy to find a lot of resource focus on the optimizing performance of websites, but there is little resource focus on the optimization of web games, especially for HTML5-based web game. As you know, for web games, you need more graphical operati ons and the users are more sensitive about the delay of the page and really need a quick feedback if anything changes. So performance is more important to web games than web applications.

It is quite interesting to find out how to test performance of a web game and try to optimize it. And we will mainly focus on the web games based on HTML5 by using Canvas. In order to show the optimization procedure, flappy bird will be taken as a case study to introduce our way of optimization performance and through this study, in order to find some general suggestions that can help you build high performance web games.

1.2 Aim

The ambition of this thesis project is to find out something that can really help us build high performance web games. It is clearly that there is always something that can be improved to our project. Usually, the process would be like this: First try to test the game and then make some improvement, and the test it again. Through this optimization procedure, the performance of the game will be better and better.

Another ambition of this thesis is to consider it as a good example of how the optimization procedure works. And by this study,a list of suggestions that can be taken through generally web games.To be specify, there are several aims of this thesis project:

1. To give the procedure of optimizing performance of flappy bird as a case study
2. To give suggestions of how to build high performance to general web games.

1.3 Research questions

By the motivation and aim above, our research are mainly focus on two parts, one is the optimization procedure through the case study of flappy bird, and another is the general suggestions of web games with high performance.

According to our aim, we figure out several interesting research questions:

1. How to optimize a web game based on HTML5 Canvas?

   The most important performance metrics are computer performance and render performance, for computer performance, it means the JavaScript computation,which also including algorithmic complexity and memory usage, and for render performance, it means the time cost on changing the data from numbers to pixels on computer.

2. How the optimization procedure works during the case study of flappy bird?
3. What you should take care of when you develop a web game with high performance?
4. How to use Chrome to test performance of your code?

1.4 Delimitations

This thesis is mainly talk about web games that based on HTML5 Canvas, and for the performance test, it is mainly based on the Google Chrome browser.

2 Theory

This chapter is used to provide to some technique background and theory for this thesis, so it will first introduce different performance metrics on web application, and then focus on the render performance and compute performance, and then different web performance test tools and the process of optimization will be clearly identified.

2.1 performance metrics

For web application, performance is particular important to user experience since the application become more and more complexity and users become more and more picky.[7] Performance can a be very widely topic and hard to measure,in order to defined the exactly performance metrics of web application[8], we usually view it from three pillars: Network, Compute and Render[1].

Network performance is the base of every performance strategy, there is nothing we can do before receive the resource. It is hard and impossible to make sure the network will always be perfect as expected, what we can do are mostly about compress the files, reduce the request times.

Compute performance is always related to JavaScript computation for web application, which including algorithmic complexity, memory usage patterns, and unoptimized code. To optimize algorithm can be an eternal topic and always play a critical role not only for web application but also for all the procedure. Memory usage in browser is always misunderstanding and can easily cause memory leak. Although JavaScript will automatically collect garbage based on references[6], but if the code is written carelessly and come cycle, which means two objects are created and reference on another, it can cause problems.

Render performance means the time cost on the transformation from data to pixels on the screen. Except the network part, render cost most of the waiting time before user get the response from the website. Whenever users do something related to the display of web page, try to give the browser the least amount of work to update the pixels on the screen, which means you should reduce both render size and complexity of painting operations as much as possible. Reduce render size means you should just update the change part rather than the whole page, and reduce complexity of painting operations means you should choose the cheapest way to paint your elements.

This thesis will mainly focus on the compute and render performance.

2.2 Performance Test Tools

Different performance aspects will use different performance test tools. This thesis will just focus on the Chrome browser and will only use the tools provided by Chrome.Specifically, we will use just Profiles panel and Timeline panel in Chrome DevTools.

Timeline panel provides an overview of where time is spent loading up your web application such as how long it takes to process DOM events, render page layouts or paint elements to the screen which gives you a record for each even that happens, displayed in a "waterfall" graph.[2] It allows you to drill down into three separate facets that can help discover why your application is slow: Events, Frames and actual Memory usage.

Profilers show us which functions take the most time, there are three types of profileing: JavaScript CPU profile, CSS selector profile and Heap snapshot[3]. CPU profiles show where execution time is spent in your page's functions. There are two views top down and bottom up. The Top Down view shows an overall picture of the calling structure while the bottom up give a list of functions by the impact of performance.

2.3 Performance Optimization Process

When optimize performance of an application, it is normal to start with the test and find the weakest part, and after optimize this part, test again and find the weakest part of the new version. After several times of this iteration, the performance will reach the requirement of users.

2.4 Related Work

There are some papers related to the user experience and performance of web application. Some resources just discuss the performance from a wide range and not provide how to optimize performance[5]. Some papers just give some ideas to optimize web application, Some books give advise of how to develop a high performance web game[4]. There do exists some good websites talk about what should be concerned when implement web games and how to optimize web games[9], but these articles are quite messy and not rigorous. And we don't find any papers focus on how to optimize performance of web games.

3 Method

In this chapter, the method is described in a way which shows how the work was actually carried out. The description must be precise and well thought through. Consider the scientific term replicability. Replicability means that someone reading a scientific report should be able to follow the method description and then carry out the same study and check whether the results obtained are similar. Achieving replicability is not always relevant, but precision and clarity is.

Sometimes the work is separated into different parts, e.g. pre-study, implementation and evaluation. In such cases it is recommended that the method chapter is structured accordingly with suitable named sub-headings.

4 Results

This chapter presents the results. Note that the results are presented factually, striving for objectivity as far as possible. The results shall not be analyzed, discussed or evaluated. This is left for the discussion chapter.

In case the method chapter has been divided into subheadings such as pre-study, implementation and evaluation, the result chapter should have the same sub-headings. This gives a clear structure and makes the chapter easier to write.

In case results are presented from a process (e.g. an implementation process), the main decisions made during the process must be clearly presented and justified. Normally, alternative attempts, etc, have already been described in the theory chapter, making it possible to refer to it as part of the justification.

5 Discussion

This chapter contains the following sub-headings.

5.1 Results

5.2 Method

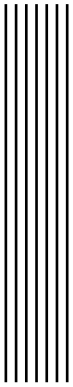
The discussion shall also demonstrate an awareness of methodological concepts such as replicability, reliability, and validity. The concept of replicability has already been discussed in the Method chapter (3). Reliability is a term for whether one can expect to get the same results if a study is repeated with the same method. A study with a high degree of reliability has a large probability of leading to similar results if repeated. The concept of validity is, somewhat simplified, concerned with whether a performed measurement actually measures what one thinks is being measured. A study with a high degree of validity thus has a high level of credibility. A discussion of these concepts must be transferred to the actual context of the study.

5.3 The work in a wider context

6 Conclusion

This chapter contains a summarization of the purpose and the research questions. To what extent has the aim been achieved, and what are the answers to the research questions?

The consequences for the target audience (and possibly for researchers and practitioners) must also be described. There should be a section on future work where ideas for continued work are described. If the conclusion chapter contains such a section, the ideas described therein must be concrete and well thought through.



Bibliography

[15a] *Html5 Feature Performance*. Dec. 2015. URL: http://www.html5rocks.com/en/features/performance.

[15b] *Performance profiling with the Timeline*. Dec. 2015. URL: https://developer.chrome.com/devtools/docs/timeline.

[15c] *Profiling JavaScript Performance*. Dec. 2015. URL: https://developer.chrome.com/devtools/docs/cpu-profiling.

[Bur13] Evan Burchard. *The Web Game Developer's Cookbook: Using JavaScript and HTML5 to Develop Games*. Addison-Wesley, 2013.

[Jay12] Mike Jones Jay Bryant. *Pro HTML5 Performance*. Apress, 2012.

[Osm15] Addy Osmani. *Writing Fast, Memory-Efficient JavaScript*. Dec. 2015. URL: http://www.smashingmagazine.com/2012/11/writing-fast-memory-efficient-javascript/.

[Pau99] Mark Crovella Paul Barford. "Measuring Web performance in the wide area". In: *ACM SIGMETRICS Performance Evaluation Review* 27 (1999).

[Pax96] Vern Paxson. "Towards a Framework for Defining Internet Performance Metrics". In: *Proceedings of INET '96, Montrea* (June 1996).

[Tea13] Agile Support Team. *Web applications Performance Testing Metrics*. 2013. URL: http://www.agileload.com/agileload/blog/2013/02/18/web-applications-performance-testing-metrics.