

Perttu Hotakainen

# Publishing Video Online with Open Source Codecs

---

Metropolia University of Applied Sciences

Bachelor of Engineering

Media Engineering

Thesis

13 May 2014

Author Title	Perttu Hotakainen Publishing video online with open source codecs
Number of Pages Date	30 pages 27 April 2015
Degree	Bachelor of Engineering
Degree Programme	Media Engineering
Instructor	Jonna Eriksson, Senior Lecturer
<p>The subject of the thesis is open source technology and its utilisation in video publishing online. The objective of the thesis was to analyse whether or not open source codecs could produce a video good enough to compare with commercial codecs, when it comes to quality and file size. Moreover, the thesis describes a trailer that was made for a game for a Finnish indie game developer.</p> <p>First the thesis goes through some terms and techniques that belong to the field of study, the history of open source code and software, and their growth among information technology.</p> <p>Secondly, thesis describes how the pre-production of the trailer was carried out with open source software VirtualDub, and how the actual editing, compressing and exporting with Adobe Premiere Pro CC 2014, from Adobe Systems were done.</p> <p>The open source codec VP8 from Google and On2 Technologies was found to be on the same level with the commercial H.264 codec in both quality and file size. In conclusion the thesis recommends and encourages the use of open source software for the sake of innovation, software environment, and supporting free video compression services.</p>	
Keywords	Open source, codec, vp8, vp9, video production, video compression, video encoding,

<p>Tekijä Otsikko</p> <p>Sivumäärä Aika</p>	<p>Perttu Hotakainen Videon julkaiseminen verkossa avoimen lähdekoodin koodekeilla</p> <p>30 sivua 7.4.2015</p>
<p>Tutkinto</p>	<p>Bachelor of Engineering</p>
<p>Koulutusohjelma</p>	<p>Media Engineering</p>
<p>Ohjaaja</p>	<p>Lehtori Jonna Eriksson</p>
<p>Insinööriyön tarkoituksena oli avoimen lähdekoodin videonpakkaustekniikan hyödyntäminen internetvideon pakkaamisessa. Työn tavoitteena oli selvittää, voiko avoimen lähdekoodin koodekeilla tuottaa yhtä hyvää tai jopa parempaa laatua kuin kaupallisilla koodekeilla. Tätä selvitettiin käytännössä mainostrailerituotannossa, joka tehtiin insinööriyön osana. Traileri tehtiin suomalaiselle pelialan yritykselle.</p> <p>Insinööriyön aikana selvitettiin H.264-standardin kehityskulkua, ensimmäisestä MPEG-1-koodekista nykyaikaan ja vapaan lähdekoodin ohjelmiston kehitystä 2000-luvulla. Traileri pakattiin H.264- ja VP8-koodekeilla, jotka olivat tekoaikaan ryhmiensä edistyneimmät koodekit. Trailerin alkutuotanto toteutettiin avoimen lähdekoodin ohjelmalla, mutta videon editointi ja pakkaus tehtiin kaupallisella ohjelmistolla.</p> <p>Avoimen lähdekoodin VP8 koodekki todettiin käytännön tasolla sekä tarkkuudeltaan että pakkaustehokkuudeltaan samanlaiseksi kuin kaupallisen puolen standardin H.264-koodekki. Opinnäytetyön tulosten pohjalta suositellaan avoimen lähdekoodin käyttöä innovatiivisen ohjelmistoympäristön ja ilmaisen videonpakkauspalvelun tukemisen vuoksi.</p>	
<p>Avainsanat</p>	<p>avoin lähdekoodi, koodekki, vp8, h.264, videonpakkaus, video- tuotanto</p>

## Contents

Abstract

Abbreviations

1	Introduction	1
2	Video Online	2
3	Compression	7
3.1	Lossless Compression Methods	7
3.2	Lossy Compression Methods	8
3.3	Compression in Action	8
4	Open Source	10
4.1	What Open Source Is	10
4.2	Google and the Growth of Open Source Software	12
5	Codecs	15
5.1	Standards, Containers and Codecs	15
5.2	Commercial Codecs	15
5.2.1	MPEG-1	16
5.2.2	MPEG-2	16
5.2.3	H.264 / AVC / MPEG-4 Part 10	17
5.3	Open Source Codecs	17
5.3.1	OpenH.264	17
5.3.2	Theora	17
5.3.3	VP8	18
5.3.4	VP9	18
6	Trine 3 – The Trailer	19
6.1	Choosing the Codec for the Trailer	21
6.1.1	Exporting with H.264	24
6.1.2	Exporting with VP8	26
6.2	Comparing VP8 and H.264	29
7	Future Perspectives	32

8 Conclusions

33

References

35

## Abbreviations

MPEG	Moving Pictures Experts Group. A Committee that publishes international standards for video compression. (Dhanani, Parker 2013, 111.)
Mbps	Mega bits per second. A unit of bit rate, indicates how many bits are conveyed per second.
Gbps	Giga bits per second
OS	Operating system
FPS	Frames per second

## 1 Introduction

During the last hundred years, video, a form of moving still images, has made its way from movie theatres to the everyday life, all around the world. It all started from physical large film rolls, but today most videos are (?) in digital format, online and in people's pockets, and displaying and producing them depends on high technology and innovations.

The purpose of this thesis was to explain the basics of open source codecs and how and why they are used and why they should be used more and more in online video publishing. For the practical part of the final year project, I have made a video game trailer, which has been compressed for online use. The project mainly concentrates on two main codecs of the project, the commercial H.264 / AVC and the open source based VP8. The encoding of the trailer was carried out with each of the codecs and the quality of the encoding was compared in order to determine if open source technology is equivalent to the commercial one when it comes to the quality and size of the video file.

The thesis is done for Metropolia University of Applied Sciences to provide an example of open source codecs in video production, to provide/give basic information of codecs, and to discuss flaws and strengths of open source software. I will explain the basic terms of video encoding and the technology around it. My main source for information is Digital Video Compression from Peter Symes (2004), along with a few other books about video technology, and multiple different web sources.

Video production technology is developing rapidly, and keeping up with the trends and innovations is important for any video sector worker.

## 2 Video Online

Video is "the recording, reproducing, or broadcasting of moving visual images" (Oxford Dictionaries 2015).

Today video can be seen all around; in billboards, the Internet, mobile phones, as parts of lectures and news; it is not only existing in movie theatres or TVs as in the old days. The main reason why nowadays videos are everywhere is that people have succeeded to compress them so that they are easy and fast to produce and display. Even small children with mobile phones can record and produce video with a few simple clicks on their smart phones.

Originally, videos were captured on a film, like photos, and then displayed one after another, with fast speed, on a machine that stretched and projected the video on film and then on a larger screen. (Burg 2007, 326) The frame rate while playing a regular film is about 25 frames per second, which means that in one second, there are 25 pictures swishing past the projector's lens. For an analogue film this is not a problem of any kind, since there is no digital data as in bytes or bits moving anywhere. Everything is just mechanics, and therefore dependable of physical motor speed and such, but when talking about digital video, the problem is noticeable. The main difference of the film from the past and video today is the capturing method; today basically everything is captured on digital sensors rather than actual film. A digital sensor captures everything it can, and transforms the light through electricity to numbers, in other words bits and bytes. This creates quickly a problem about how to keep the incoming data low enough to be transported between machines and to be manageable. In fact, in today's world, handling and moving information is one of the largest concerns in technology. (Symes 2004, 2.)

Digital video is built from pixels and is normally divided in to resolutions (pixel x pixel matrices). Overall, there are several definitions for the resolution of the video, such as standard (SD), high (HD) and ultra-high definition (UHD). As figure 1 shows, these examples define and stand for the pixel quantity inside the screen. For example, SD has about 300,000 pixels, whereas HD contains a little over two million.

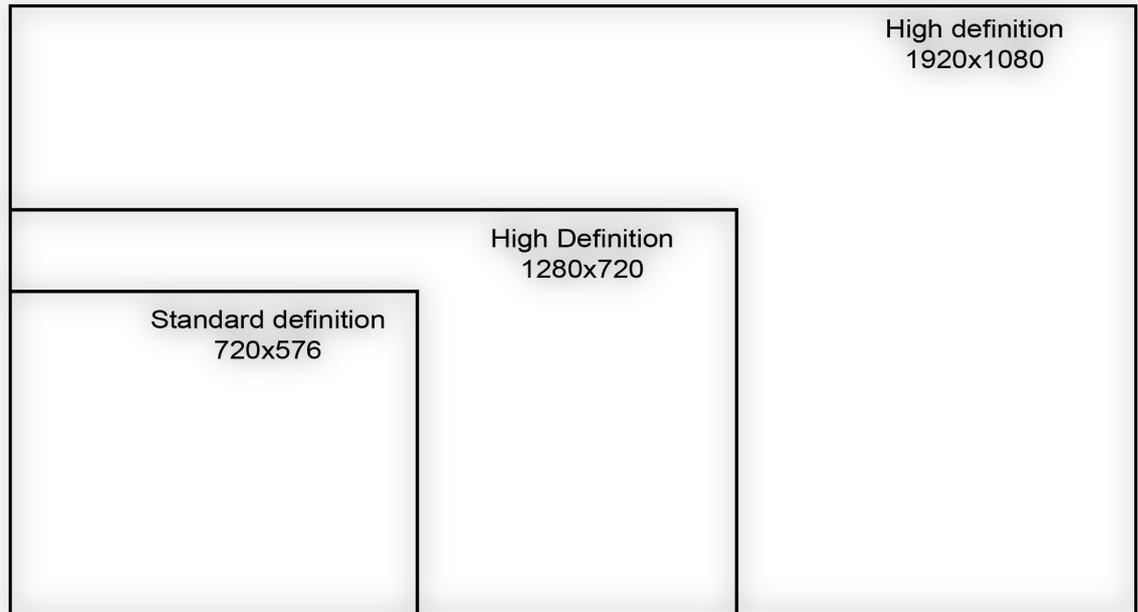


Figure 1. Different frame sizes

Each pixel consists of three colours – red, blue and green – and how much each colour gives the pixel its value. The value of the pixel then leads to the size of a single video frame in bits. (Dhahani 2013, 5-7.)

Available bits express the value of the pixel, and the more bits a machine can use, the more accurate a picture it builds. For example with 8-bits a picture can have 256 different values for red, green and blue each ( $2^8$ ). With 8 bits each pixels needs 3 times 8 equals 24 bits to have its value. HD-video has 1920 by 1080 pixels, which means roughly two million pixels and therefore 24 times 2 million equals 48 million bits per one frame, as shown in figure 2, (Dhanani 2013, 8-9.)

FORMAT	HORIZONTAL	VERTICAL	DISPLAYED PIXELS PER SECOND (25FPS)	BIT RATE (8-BIT)
SD	576 pixels	720 pixels	10368000	248 Mbps
HD	720 pixels	1280 pixels	23040000	550 Mbps
FULL HD	1080 pixels	1920 pixels	51840000	1,2Gbps

Figure 2. Sizes and bit rates of different video formats. Data gathered from Dhanani (2013)

When this is multiplied with a speed of a regular 25 frames per second in film cameras, it brings the size to an enormous amount of 1.2 billion bits per second. This means that regular Full HD-video, if not compressed, needs 1.2 Gbps worth of bandwidth to move from one device to another. (Dhahani 2013, 8-9.) An average household globally has a connection of 22.1Mbps (Ookla 2015). As figure 3 shows, the fastest nation in the world, Singapore, has an average broadband speed of 111.56 Mbps (Ookla 2015), which is still less than 10 % of the bandwidth speed needed for transferring a regular quality, raw, uncompressed HD-video. Needless to say, video needs to be compressed into a smaller format in order to transfer and play it with regular devices. Codecs (en-code/decode) have been invented to make this happen.

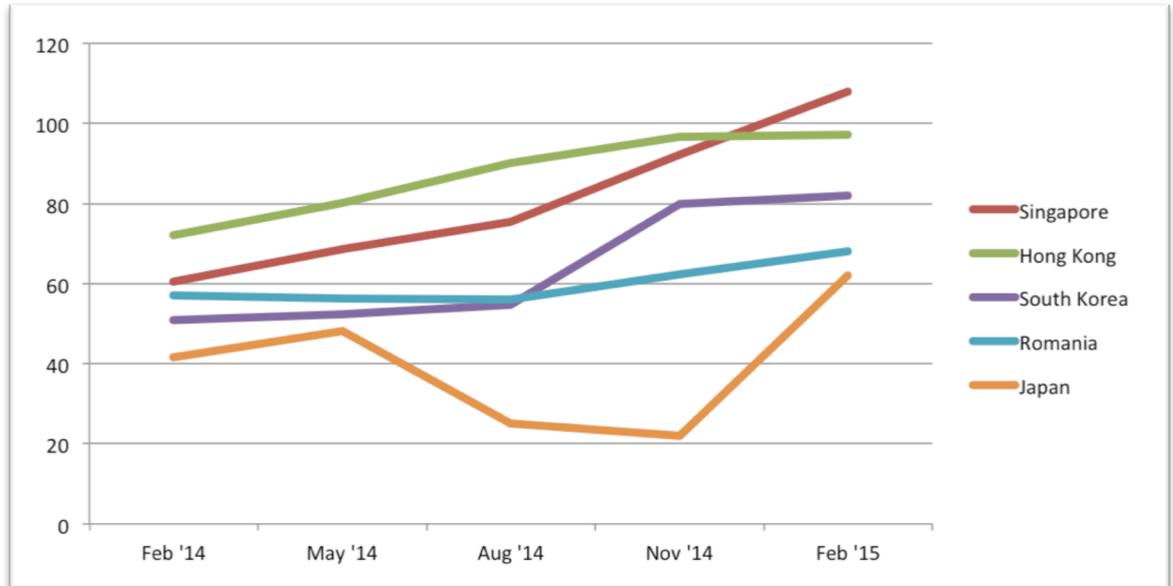


Figure 3. Average speeds of the top 5 countries with fastest broadband speeds. (Ookla 2015)

Codecs are algorithms and/or programs used to compress and decompress video and audio. Nearly all of the video that exists is compressed because raw video is unusable due to its large size. Video cameras have in-built systems for compressing video whilst recording so it can be saved on a memory card and transferred onwards.

Digital video cameras today usually have multiple codecs to choose from. Depending on the price range and target of the camera – whether it is a professional movie camera or an amateur camera – the codecs vary from the basic H.264 to Apple's ProRes.

The Canon EOS C100 (seen in figure 4), which is a Cinema EOS camera production line from the camera manufacturer Canon, produces 60 fps video that is compressed with the H.264 codec, and a ready made video has 24 Mbps, full HD quality.



Figure 4. Canon Eos C100 Video camera (Canon 2015)

24 Mbps is a lot, and takes a lot of power and broadband speed if it is wanted to play it immediately from the camera, but it is possible today, however. The Blackmagic production camera on the other hand has a 4K resolution, and provides the user a choice between using the ProRes 422 codec from Apple, or a (visually) lossless compressed CinemaDNG codec. The Blackmagic camera, when all the settings are set up at maximum, produces a data rate of 880 Mbps. This is of course not suitable for web streaming, or playable in any consumer media, but it also is not meant to be used in amateur production.

In the following chapters the thesis will explain more about codecs, lossy and lossless compression and workflow when producing video online.

### 3 Compression

There are two types of compression, lossless and lossy. The difference between them is that in lossless compression, when decompressing, all the data compressed is being retrieved because only redundant data had been removed. The data removed for compression should then be able to be recreated from the remaining pieces. This causes the lossless compression to be significantly higher in file size, so most of the video is being compressed with lossy methods. However, both of the methods can also be used for the same file, in a case when a video is first compressed with the lossy method in a video file, and then compressed again with lossless methods for transfer to make the file size even smaller. (Symes 2004, 4-5.)

#### 3.1 Lossless Compression Methods

There are generally two lossless techniques in use, run-length encoding and entropy encoding. In graphical images in run-length encoding, long runs of the same value can be expressed in a lot shorter method. For example when several pixels have a certain, exactly the same value, it takes a lot less power to code pixels inside one, new value, in stead of repeating the same value over and over again throughout the run. (Symes 2004, 6.)

Entropy coding is much more complex than run-length, and is usually used in the last step of a compression scene. It is based on an idea that all the symbols don not need to be of the same length. As long as both sides of the transaction know the rules of the code used, it does not matter if a certain symbol stands for one or 200 symbols. Entropy coding is compressing the data by representing frequently occurring values with short symbols, and therefore in case of the input data not being random throughout the data code, the number of total data will be smaller than the original values. The basic idea is not new, and it is used for example in ASCII encoding system, where any letter or number is represented by a particular set of one byte. The value of the letter does not matter; the symbol is always eight bits (one byte) long. In multimedia the biggest problem with lossless compression, in addition to the bad compression ratio that produces too large files, is that it also does not guarantee a fixed bit rate, which is needed when transferring music and video. Some parts of music and video are always too

complex to be compressed and this makes it impossible to have lossless compression with fixed bit rate. (Symes 2004, 5-7.)

### 3.2 Lossy Compression Methods

Because of the problems of the lossless methods, lossy compression is normally the choice made when working with multimedia. The ideal situation is that in lossy compression it would be possible to only compress or remove irrelevant data, so that when the product is decompressed, data that is forever gone would not have an effect on the end product. This of course is not usually the case, and also a noticeable amount of data is missing. Lossy compression still aims for a simple goal: maximum compress ratio, or reduction of bit rate with minimum cost, which in this case is lost data or loss in quality. This of course is not simple and there are many different codecs in the digital world. Some of them are commercial products, for which the software companies have to pay, and some of them are free, open source codecs.

### 3.3 Compression in Action

It is important to ask what actually happens when a video is being *compressed*. It depends on the codec and how complex it is. In early years, when MPEG-1 was still in use (MPEG-2 was standardized in 1995) each picture or *frame* of a video was analysed and compared to the next and previous frames. (Symes 2004, 153)

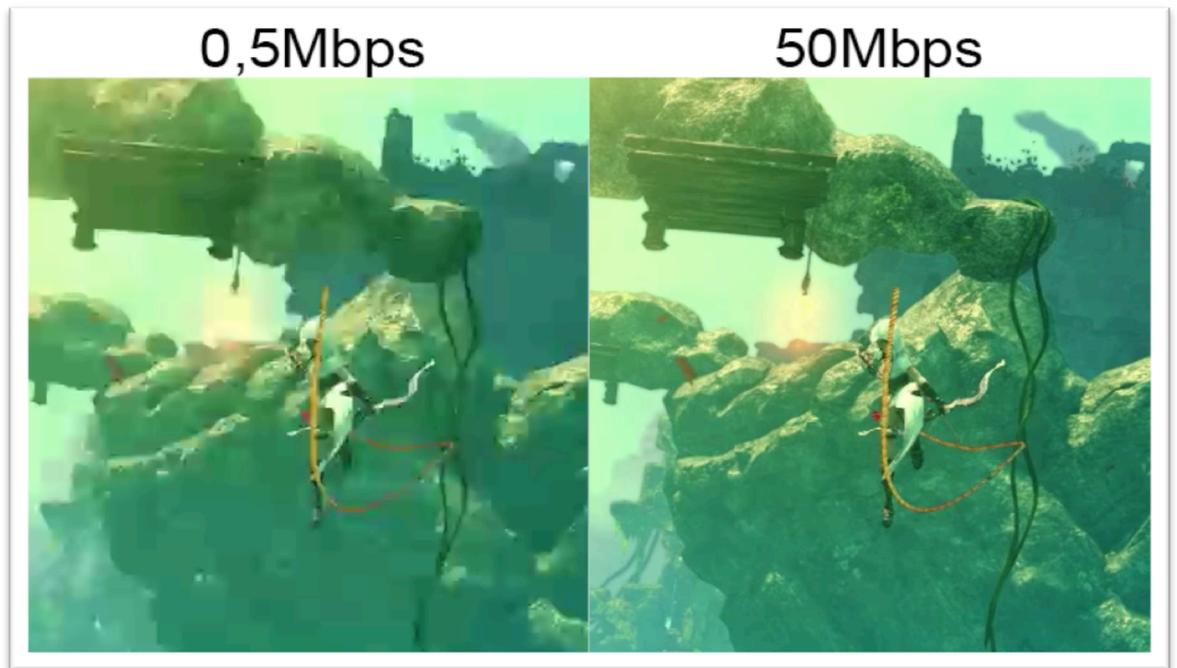


Figure 5. Screenshot of different bit rated video. (VLC 2015)

That original idea still continues from those days. Frames of the video are compared to the previous and checked what information is preserved, what parts are new. As figure 5 shows, the difference between highly compressed and less compressed quality is clearly noticeable – the picture on the right in holds much more detail due to the higher bit rate. Based on these original ideas, if multiple highly mathematical algorithms, are followed it is possible to remove data but still keep the video sharp. (Symes 2004, 152-154)

## 4 Open Source

### 4.1 What Open Source Is

Open source stands for a software or development method where users have free access to the source code of the software or process in question. It is based on voluntary work, when users contribute their own free time to develop the code. (Lakhani 2002.) Nowadays open source software can be found in nearly all the fields of information technology. For example operation systems based on Unix, which is an open source project, are estimated to be used in roughly 67.8 % of *all* the web servers in the world. (W3Tech 2015) Android, which holds over 74 % of mobile markets shares when it comes to the operation systems in phones and tablets, is an open source operation system. (NetMarketShare 2015)

Why are open source methods so popular? Open source methods have risen from the legacy of the free software movement from the early 1980's, started by Richard Stallman who founded the Free Software Foundation (FSF), opposing the rising commercial software companies. The key idea was to have all the code open, so that anyone who had the skills and enthusiasm for developing software further or modifying it to his or her means had a chance to do so. (Lakhani 2002, 3)

Apache works as an example of a functioning open source product and community. Apache software is used on web servers, which connect people to the Internet. The main function of such a server is to request and deliver information from the servers to the end users' browser. Since Apache is an open source program, anyone can alter the code, produce better functionalities and evolve the software. Apache does not have its own official support staff, but a great number of individuals who solve problems people are having and who provide useful information for people starting to use the server. This community driven approach is what makes open source software and people around it so special popular. As figure 6 shows, between 1995 and 2000, just inside five years, web pages hosted in apache driven servers rose from zero to over nine million. This is mainly thanks to the open source community, without depreciating the usefulness of the main founders of the company, of course. (Lakhani 2002, 930)

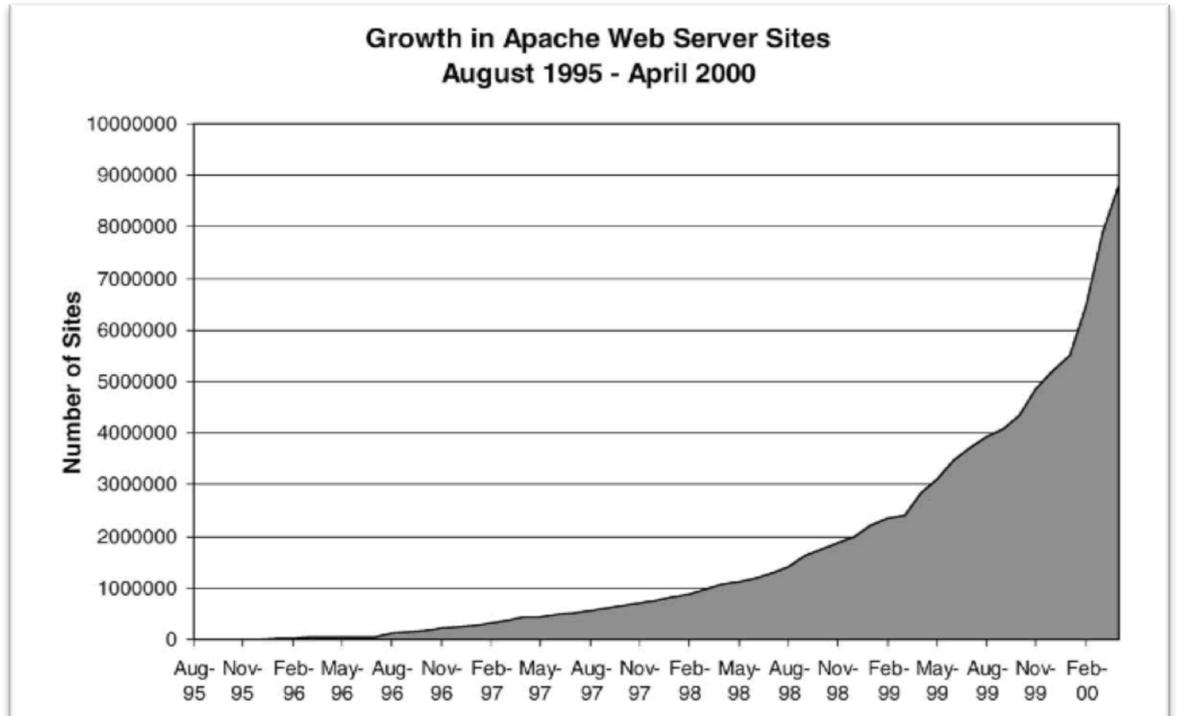


Figure 6. Growth in Apache Serves between 1995 and 2000. Reprinted from Lakhani (2002).

Open source has its opposers, who mainly come from the business side, and from patent organisations such as the MPEG-LA (an intellectual property management organisation that holds the patents pools, not to be mixed up with MPEG, regardless of the name) who makes its best to stop open source video codec projects for developing, mainly because they meddle with the interests of the companies it represents. Companies that own licences in MPEG-LA patent pool include large international hardware and software producers, such as Apple Inc, GE Video production, LG Electronics and Microsoft. (MPEG-LA, 2015.)

Naturally all this means that in the world of codecs, whereas commercial codecs like H.264 represent patent protected technology, there is an equivalent free project in the open source communities. Most noticeable of the open source codecs are the VP8 / VP9 that are developed by Google and On2 Technologies. (WebM 2015)

## 4.2 Google and the Growth of Open Source Software

Google, despite being one of the largest (if not the largest) software companies in the world, is actually also one of the main supporters of the open source technology. Android is easily the most used mobile phone operating system in the world, with over 76% share of the smartphone markets. (IDC 2015) Even though Google is behind the Android OS (operating system), all the manufactures themselves can alter the smartphone OS for their needs. There is also a great amount of community designed and coded Android operating systems in online, where basically anyone can participate in making and developing the systems. The open source communities are usually not involved with any of the actual smartphone manufacturers, but can be downloaded to most of the most common smartphones, and installed by users themselves.

As Google is one of the largest supporters of Open Source software, it is only natural that it is also a great supporter of open source video and codecs, and tries to promote them in software it produces. Google Chrome is notably the most used web browser today. It surpassed Firefox from Mozilla in March 2012, and has been growing ever since, and over 62% of all the people online use it regularly today. (Ookla 2015) As Figure 7 shows, open source software has over 85 % of the Internet browser markets.

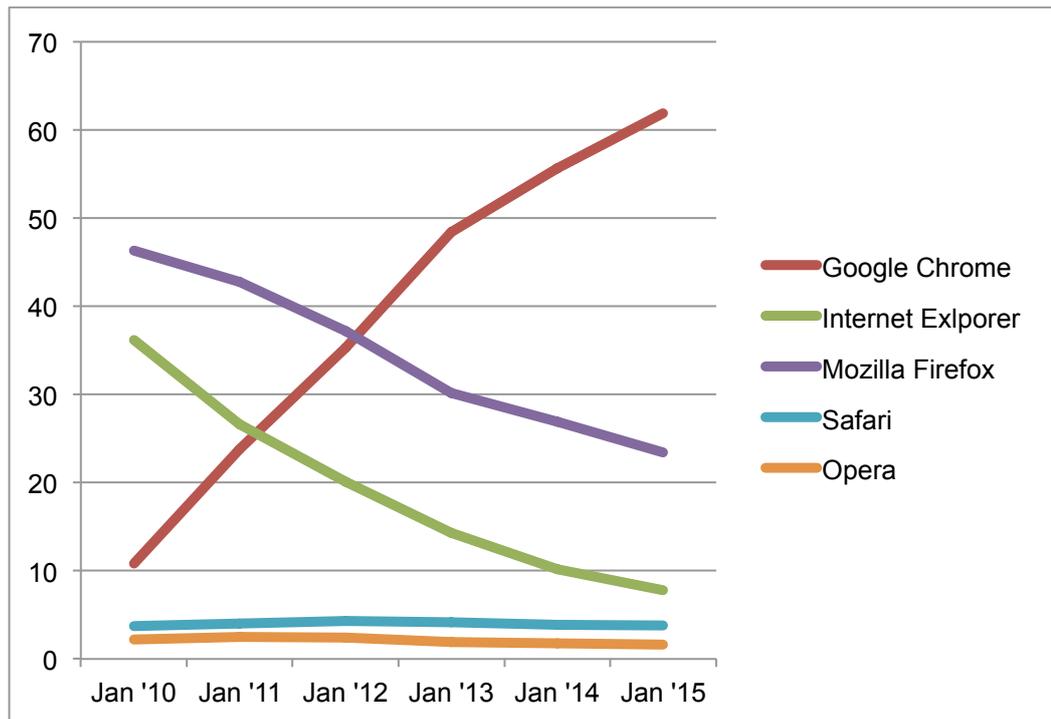


Figure 7. Internet Browsers Market Share 2010-2015. (Ookla 2015)

That is nearly two times as much as the following three browsers, Mozilla's Firefox (which is also an open source software), Microsoft's Internet Explorer and Apple's Safari combined. (W3Schools 2015.)

Google's YouTube (famous logo is seen in figure 8) is the largest video sharing website in the world with over a billion users. Every minute 300 hours of video is being uploaded to YouTube, and every day it streams hundreds of millions of hours. The usage grows 50% per year, and over half of the users are on mobile. YouTube formerly used Flash technology as its primary player for web videos, but since the start of the year 2015, the standard player was changed to first try playing with HTML5 video, which supports and promotes the usage of open source video. (YouTube 2015.) Google's own web browsers' large usage is the reason why that can be done, since today most of the browsers support the new, open source method of playing video. This method, HTML5 Video does not need an external player, but can play the video straight from the browser. External plug-ins always cause extra trouble for the user, and the ideal situation therefore is that extra settings and set-ups would be never needed.



Figure 8. YouTube™ logo. (YouTube brands 2015)

Google Chrome, Opera and Firefox from Mozilla have supported VP8 for a long time, (WebM 2015) and now when YouTube supports it too, it seems true that every software technology are heading for a more open development methods, towards open source software. (YouTubeCreators 2014)

## 5 Codecs

In daily discussions, codecs are often mixed up with containers to mean the same thing. This is due to the fact that the container is the visible part of the file, meaning the extension, such as .avi, .mov, .wmv or .mp4. Multimedia containers can hold multiple tracks of video and audio. Even though containers *might* tell something about the codec inside, they should not be confused – containers can have several different codecs. Some popular containers and extensions today are .mov developed by Apple Inc, .mp4 developed by the Motion Pictures Expert Group (MPEG) and .wmv developed by Microsoft. Inside these containers there are dozens and dozens of more or less distinguishable codecs, such as H.264, DNxHD or ProRes. (Dhahani 2013, 8)

In the upcoming chapters is explained and clarified facts and differences of some of the most common commercial and open source codecs.

### 5.1 Standards, Containers and Codecs

A standard is a document containing specifications, rules, guidelines and other information that can be used in producing different products according to their purpose. (ISO 2015.) In the world of codecs, standards hold the basic codecs that are under developed by any time. H.264 is a standard for H.264 / AVC based codecs. It is over 550 pages long, and mostly includes highly technical mathematical formulas, definitions and descriptions. It explains a method of how to encode and decode (enCOde / DECCode = CODEC) a video, and most importantly, how to do this efficiently without losing quality. MPEG standards describe different tools that help to compress, and five examples how the methods in questions might be implemented. The standard stands as a base for people that want to participate in working with the codec in question. (Richardson 2010, 5)

### 5.2 Commercial Codecs

There is a large set of commercial codecs used nowadays, but the most common today, and the usual standard is the H.264/AVC codec. Codecs are patented like any

other software, and patent infringements cause a lot of trouble between different developers.

### 5.2.1 MPEG-1

In the beginning, the task for the Moving Pictures Experts Group was to develop a standard for encoding video at bit rates small enough to be transported and replayed from CD-ROM – about 1.5 Mbits per second. That might not sound much, but when taking into consideration that by that time, a regular audio CD had a data transfer bit rate a little larger than 1.4 Mbits per second, the goal was to basically involve video together with the audio, without raising the data rate. MPEG-1 did this job relatively well. It has a simple layered system to transfer both audio and video on a stream of data, but the downside is that in order to have them synchronized nicely, they had to have the same time base. This problem is one of the improved methods in MPEG-2. (Symes 2004, 152-155.)

MPEG-1 was the first major codec and encoding standard for multimedia. Its main definition for a file is a sequence of pictures. Inside the sequence there is a *group of pictures* (GOP). When studying the structure a little bit further, GOP transforms to multiple *frames*, or pictures, which are then seen as a video. One GOP, without the compression, can then be as small as a frame, but in MPEG-1 it usually is 10 to 30 pictures long. Each frame (or picture) is then divided to *macroblocks* and *slices*, and this level is where the compression happens. Macroblocks include all the information required for the pixels to show what wanted. (Symes 2004, 152-155.)

### 5.2.2 MPEG-2

MPEG-2 could be considered to be the inspiration for all the modern major codecs today. When using the same frame rate, it is about 50 % more complex than MPEG-1. (Symes 2004, 172.) Its development process was started when MPEG-1 was finished in 1991 and it was standardized in 1995. It includes all the coding tools from MPEG-1 but with a great number of new ones. MPEG-2 was in fact so good that the developers had to stop the work on MPEG-3, because it was not going to get good enough improvements, and the developers decided to start with working with MPEG-4. (Symes 2004, 174.)

### 5.2.3 H.264 / AVC / MPEG-4 Part 10

H.264 or Advanced Video Codec is a codec published by the International Telecommunications Union (ITU-T), and developed by the Joint Video Team. It is also published by ISO/IEC organizations as MPEG-4 Part 10. H.264 is the recent industry standard. It is the most used codec due to its easily most efficient compression ratio, which is due to its complex way of handling video. H.264 is capable of handling over 50 % greater savings in bit rates than its predecessor H263v2. (Symes 2004, 227) H.264 has great improvements to its predecessors. For example, some of the macroblocks are 75% smaller, which means among other things much better quality for picture, better motion estimation for moving frames, more accurate motion vector. (Symes 2004, 245.)

The successful development of H.264 can be seen for example in that even though the standard was published in 2003, The standard is still widely used in 2015, twelve years later.

## 5.3 Open Source Codecs

As there is a variety of codecs on the commercial side, there are also a lot of open source codecs, both made by individual communities, but also run by organizations. Like all the open source software, codecs are free to use, regardless of what they are used for.

### 5.3.1 OpenH.264

Open H.264 is called a half open source codec. It is provided by Cisco, and offered free for both non-commercial and commercial projects. (Open264 2015)

### 5.3.2 Theora

Theora is one of the most well-known open source codecs available. It is developed and published by the Xiph Open Source Community and has been in public release since November 2008, which makes it a relatively old codec. It stands out as a low bit rate codec, and it is used for example in the game industry, for example Frozenbyte. It

offers easily implementable players that convert different videos to their own Theora codec and displays them inside other programs. Theora is a good codec for small video files in indie-developed software that do not want to use a lot of money for videos. (Theora 2015)

### 5.3.3 VP8

VP8 is probably the most known open source codec today. It is developed by On2 Technologies in partnership with Google, and is a free codec for HTML 5 under the BSD-licence and it is supposed to become the challenger to Flash Player's dominance in web streaming. It is also often compared to H.264, which is the most used online video codec today. Like H.264, VP8 is highly efficient and can produce a great quality video with a fast encoding time and small file size. The VP8 was chosen to be used in the final year project, and it will be discussed more later in the thesis. (WebM 2015)

### 5.3.4 VP9

VP9 which is a codecs from Google still in development is expected to be the next big thing amongst open source codecs, and supposedly far better than any other when it comes to compression ratio and quality. Unfortunately it was not yet optimized enough to implement at the time of carrying out this study.

## 6 Trine 3 – The Trailer

A trailer was done for a game called Trine 3: The Artifacts of Power for the final year project (see figure 9). The content was recorded, imported to Adobe Premiere Pro CC, cut and edited the material and finally exported it to be played in other medias. As for being one of the main marketing materials for the Frozenbyte's new game, the trailer was expected to have a maximum coverage over multiple platforms and machines to have as large audience as possible.



Figure 9. The teaser page of the Trine 3, including the embedded video. (Frozenbyte 2015)

For this purpose, there were supposed to be found codecs that supported as many media as possible. H.264 is most common codec today, and it was used in export settings, but since open source methods are more and more available and running in more devices than ever, The trailer was also decided to be exported with VP8, the latest finished open source codec from Google and the WebM project. Unfortunately VP8 encoding was not supported in Adobe Premiere Pro, but luckily a plug-in was found, that made it available.

For a setup, a separate recording computer from the playing PC was installed, that captured the main content of the trailer, with full screen video capture. That way it was managed to get the maximum quality out of the game, because then the recording itself

did not have any effect on the actual playing of the game. This was highly important because the gameplay itself had to be as stable as possible. The capturing was done by an open source, GNU licensed program called VirtualDub. It provides a choice for capturing uncompressed video in the .avi container, which suited perfectly since it was then possible to import the video to Adobe Premiere Pro and start editing right away.

Adobe Premier Pro is one of the most recognized video editing software in the world, and both amateurs and professionals use it. It belongs to the Adobe Creative Cloud package that includes many different multimedia production software, ranging from video software to audio software, and web development software to illustration software. Adobe is a multi billion dollar company, the revenue of 2014 being over 4 billion US dollars. The Creative Cloud has 3,4 million subscribers all over the world, which makes the software known worldwide. (Adobe 2015)

It is a full packet so it can be used from the first raw cutting until to the very last exporting, which makes it really good software for small budget companies. It also usually comes bundled with Adobe After Effects, which is designed to produce effects for the video, which can then be exported back to Adobe Premiere Pro. The simple way of combining postproduction with main editing and relatively reasonable pricing (€50 / month for the package containing all the Adobe made software, with company subscription) forced to the use of Adobe Premiere for the main working software.

The production was started with uncompressed AVIs, where usually each of the video clips – even though they lasted less than 10 seconds – were gigabytes in size. As discussed earlier, raw video has a bitrate so large that it cannot be used in normal situation, which is why all the video had to be encoded and compressed after it had been cut and edited. The material was imported to Premiere, where the clips were compressed more for easier viewing with different players.

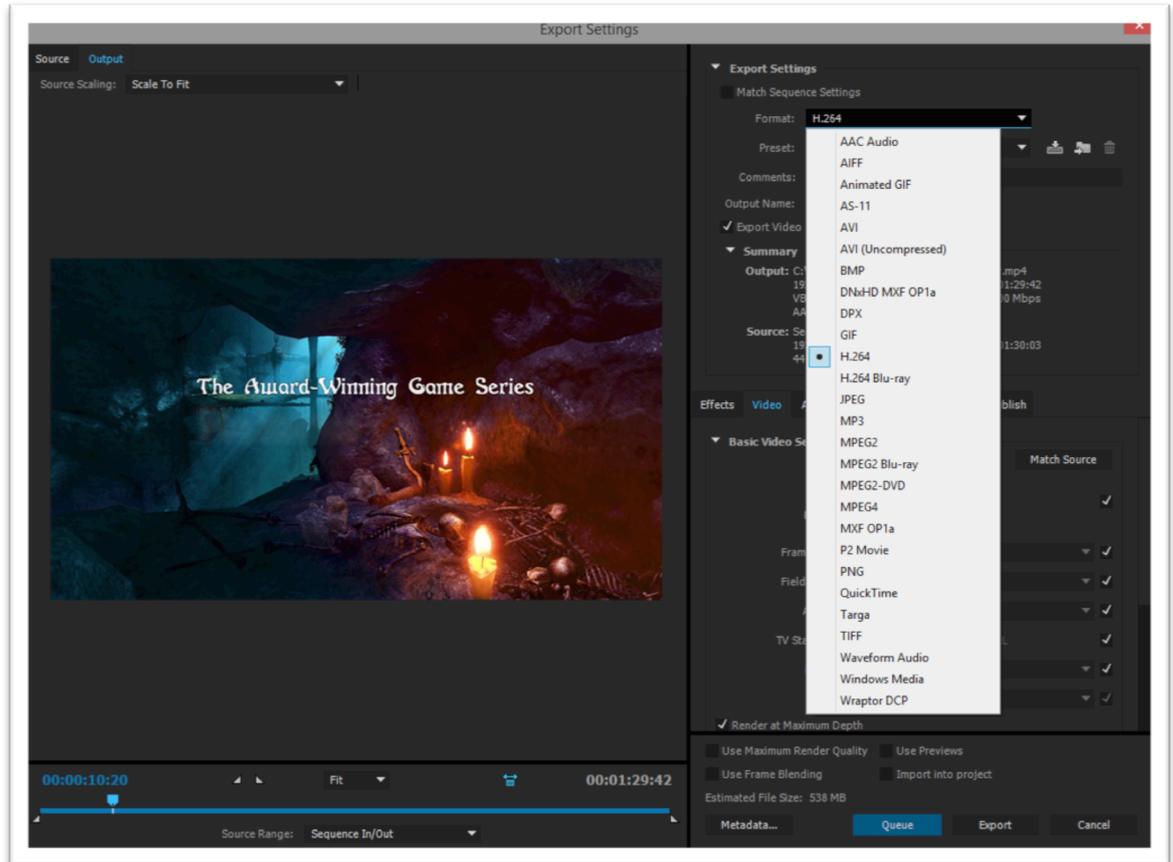


Figure 10. Different codecs to choose from when exporting from Adobe Premiere Pro (Adobe 2015)

In Premiere there are multiple codecs to choose from and several other settings to optimize the size and quality of the video.

### 6.1 Choosing the Codec for the Trailer

What is there to decide when exporting a video for maximum amount of playback devices? A Codec is the main key. Codecs make the rules on which application and software the video can be played on and which not. For the trailer, the main channel for showing the video was going to be YouTube, so for the next logical work flow step it had to be found out what YouTube requires, in terms of codecs and settings.

As discussed earlier, YouTube is the world's largest video streaming service, and has several billion unique views every day. Six billion hours of video is watched from YouTube monthly, so it is obvious that having a trailer on YouTube is the primary goal

in terms of visibility. (YouTube 2015) YouTube supports multiple kinds of containers and codecs:

- .MOV
- .MPEG4
- MP4 (H.264)
- .AVI
- .WMV
- .MPEGPS
- .FLV
- 3GPP
- WebM (VP8 & VP9)

From these, the most widely used codec needed to be chosen, which would then be MP4 and H.264, as was earlier decided.

YouTube alone was not enough for the project. Firstly, because YouTube forces in usage of their own embedded player should the user want to share the video from there (Youtuben käyttäjäehdot 2015). Secondly, since the product is a trailer for a video game and it is going to be published in multiple platforms, it was needed to have something that supports all the other medias also. Even though for most of the medias H.264 was already enough, all of the service providers were hoping for that. Some places asked for an Adobe ProRes codec, which was not available, but settled for an open source VP8 from Google, since they supposedly could easily modify that for their needs.

The best thing with the VP8 was, that whilst it is supported nearly everywhere, it promotes the user friendly, innovation-promoting business style of open source software. As earlier discussed, open source has had a successful development evolution for years, and from my opinion, it is always a good thing to support it wherever possible.

The next decision had to be made about the basic settings of the video, which can be found from in nearly all of the codecs. These settings include options such as:

- Frame rate
- Resolution
- Bit rate
- Audio bit rate

The importance of these settings is always significant, regardless of what codec is used as they describe the quality and size of the video.

Frame rate is the amount of pictures flashing on the screen in a second. Depending on the playback device, normal frame rates vary from 24 fps to 60 fps (frames per second). 60 frames per second is relatively fast, and video streaming services like YouTube had just started to supporting fast fps during the year 2015. This is due to double the bitrate needed to play the video when compared to the regular standard, 30 fps, and therefore higher costs. (YouTubeCreator 2014) For the personal experience, the higher the fps is, the smoother the video looks like, and that is the reason the trailer was exported with 60 fps.

Resolution tells how large the video screen is, how many pixels each frame contains. Today a full HD quality, 1980 pixels horizontally and 1080 pixels vertically is basically a standard in web videos. Whereas just a few years ago SD-quality was the regular resolution, today even the mobile phones can have a full HD resolution. (Samsung 2015) Therefore a full HD resolution was needed for the trailer of the final year project, and the chosen codec was expected to be able to have a smooth and detailed ability for high resolution encoding.

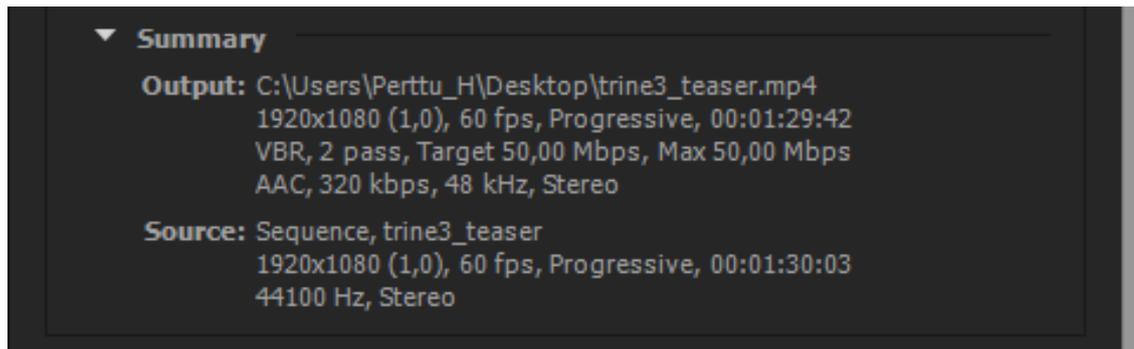


Figure 11. Settings for the trailer. (Adobe 2015)

Bit rate basically defines the quality of the video. Larger resolutions need a larger bit rate in order to fill the bit depth needed per pixels, and the higher the better. For a maximum quality, a bit rate of 50 Mbps was chosen, in order to include as much colour depth and details as possible. This does however produce a fairly large file (around 500 megabytes) for a one and a half minute long trailer, but since it is mainly going to be streamed from the internet and different video stream services, it was important to provide the trailer as high quality as possible within reasonable limits.

Audio bit rate defines the quality of audio track – the higher, the better usually – but for web use, 320 kbps or even 198 kbps is enough. Audio bit rate is also not causing a noticeable difference in the size of the file, so there is no reason to scale it down in a normal situation. Some publishers however want a lower quality audio, but this is luckily very easy to re-render, since it is only a matter of one setting, and does not affect anything else. After deciding which codecs and setting were going to be used, it was time to start exporting the video.

#### 6.1.1 Exporting with H.264

H.264 was the first natural choice for the codec for the final video as it has an excellent quality to size ratio, and is supported nearly everywhere. The trailer needed to look as good as possible in order to sell the game, so a high bit rate for the video was needed. It was also exported with 1080p full HD resolution for maximum size. (See figure 11, and figure 12.)

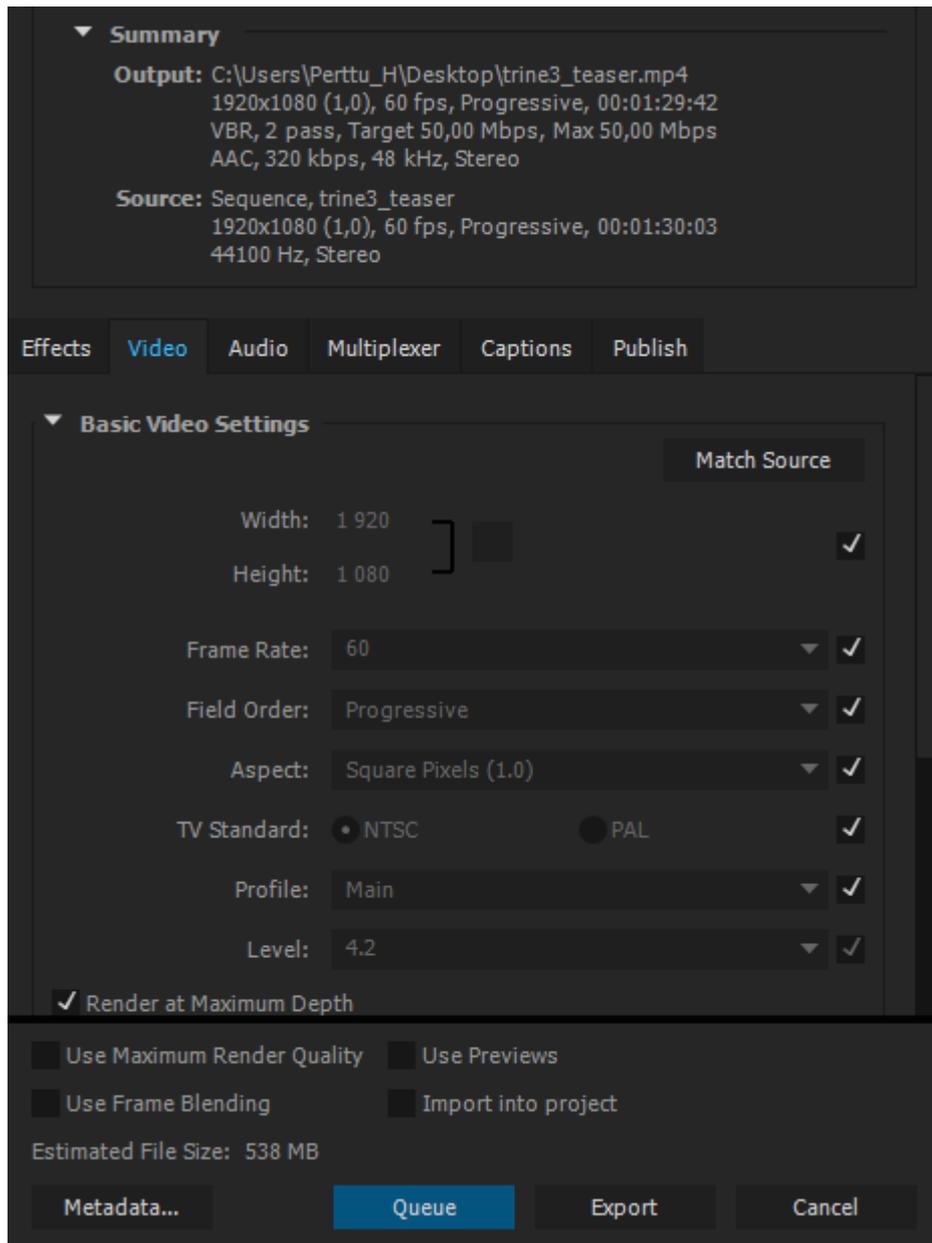


Figure 12. The settings for H.264 codec. (Adobe 2015)

H.264 was relatively easy to work with. Since it is so far developed, it has easy settings where to choose from, and wide bit rate possibilities for the quality. It was also possible to render out a video with 200 Mbps, but that produced a file over 2 Gb, and would have been too large for playing in regular players. It has a choice for either VBR 1 or VBR 2 pass, which stands for how many times the video is analysed before being exported out. VBR 2 produces better quality, but also takes a little longer to render, which in this case was not really a problem.

VBR 2 analyses the video twice, so it knows precisely what needs to be compressed and what can be used from previous frames. This works out as better gradient colours and better quality of video.

H.264 also has a choice for either progressive or interlaced field order. This means that the exported video can either be displayed as a one big frame, or as a frame that is divided into an upper and lower field. In interlaced video, the lower and upper field of frame update the picture alternately. This causes the frame rate to double, without a significant raise in bit rates, but might also cause screen tearing (vertical lines in video, where the picture is not displayed correctly) and other visual problems. However, because this allows better bit rate to frame rate ratio, it has become the usual broadcast method for PAL and NTSC analogue television broadcasting systems. (Dhahani 2013, 12)

#### 6.1.2 Exporting with VP8

Before finally selecting the VP8, the latest codec from Google still in development, VP9, was tried. It was already supported on YouTube, with Ultra High Definition 4K resolution (YouTube 2015) (only with the Google Chrome browser) but as we did not have the material or need to produce so large video and since VP9 was not yet fully optimized, it turned out that it was not necessary or ready for real use yet. When trying to render with VP9, the video was left to render for over 12 hours. However, after 12 hours, only 14 % of compressing was done (see figure 13). This was obviously too slow for the project's purposes, so the newest generation of open source codecs was abandoned and the project concentrated on the stable version.

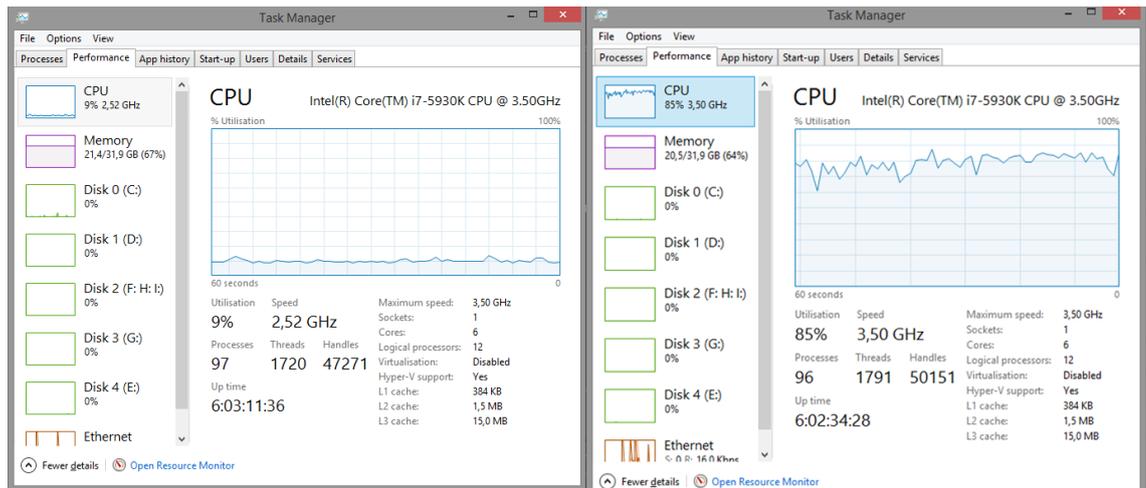


Figure 13. Comparing the functionality of VP9 (on the left) to VP8. VP9 was not able to take advantage of the hardware power the computer had because of the poor optimization. (Microsoft Corporation 2015)

As mentioned earlier, Adobe Premiere Pro does not natively support the WebM codecs, so for that purpose an external plug in had to be found in order to use Premiere Pro for encoding. After a search, it was decided to use a BSD-licensed, open source plug in from an amateur programmer named Brendan Bolles. The encoder was easy to setup – it only needed a downloaded file that was then copied into the plug in folder of Premiere Pro. After the process, the codec appears on the list among other codecs, as can be seen from the figure 14. (GitHub 2015.)

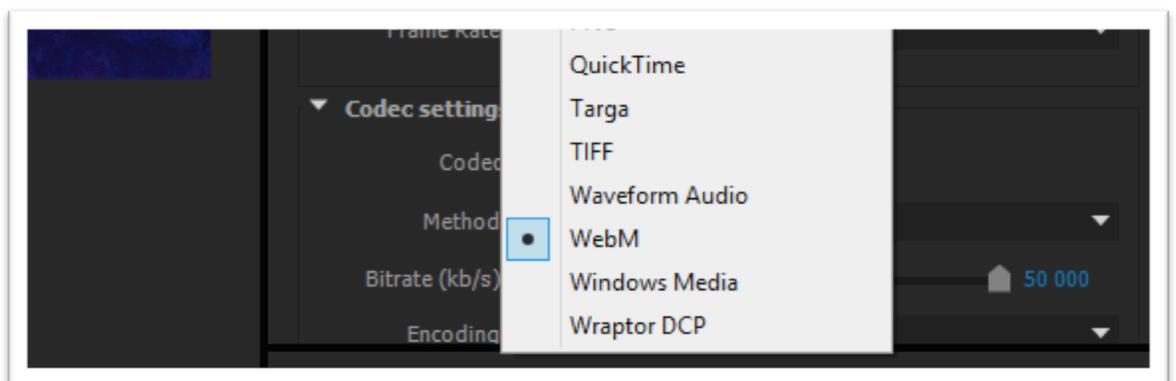


Figure 14. WebM showing on the list of codecs in Adobe Premiere Pro. It includes a choice for both VP8 and VP9 in the settings. (Adobe 2015)

VP8 had slightly different settings to choose from while exporting when compared to H.264. (see figure 15) Instead of having only the basic video settings and bit rate settings, VP8 has a few profiles to choose from that define more accurately how the video

is compressed. Encoding is divided into regular, good or best (in this trailer it was chosen to be best, as can be seen in figure 15) that change the settings for sub sampling and bit depth. On the other end H.264 has a much higher scale for bit rate, when the bit rate goes up to 300Mbps, whereas VP8 is limited to 50Mbps.

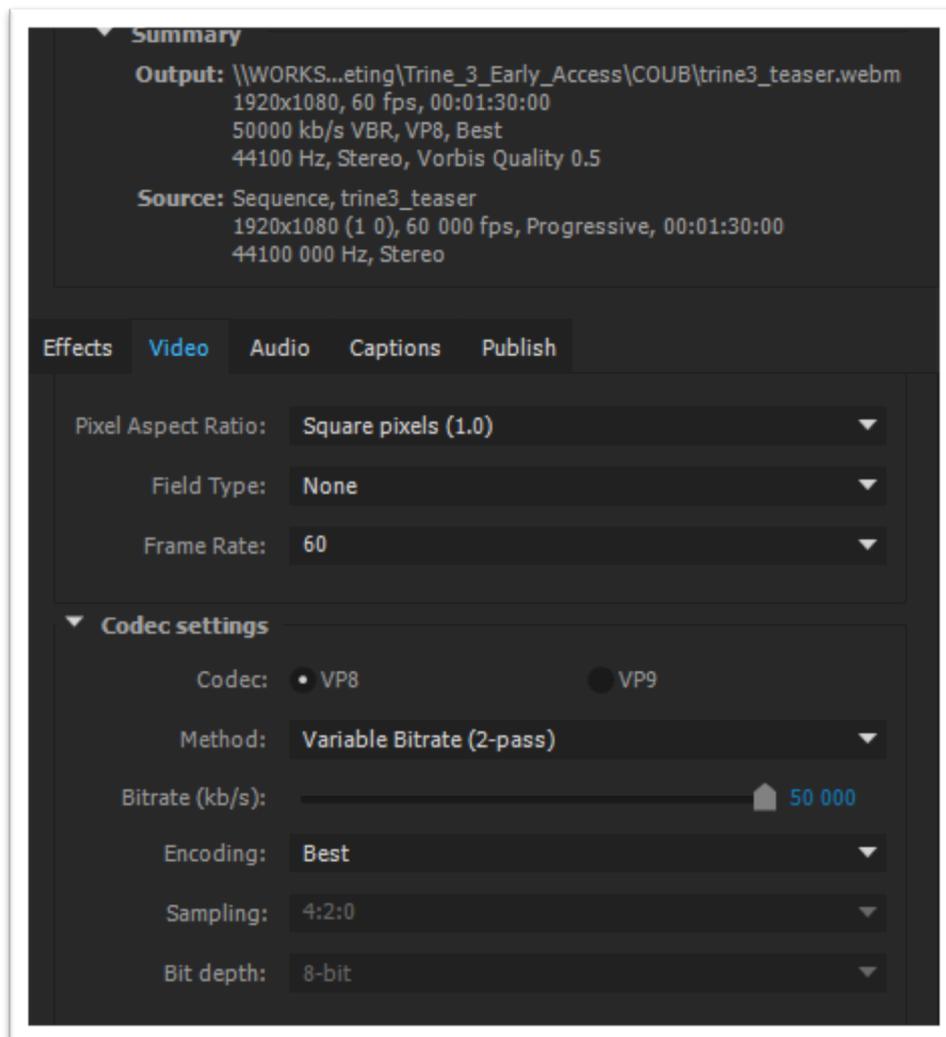


Figure 15. Settings for the WebM VP8 codec. (Adobe 2015)

## 6.2 Comparing VP8 and H.264

When set side by side, there is not much difference with the two main codecs of commercial and open source techniques. Both of the codecs took about 30 minutes to render the trailer, and there is a quality difference mainly in colour, as seen in the figure 16 below.

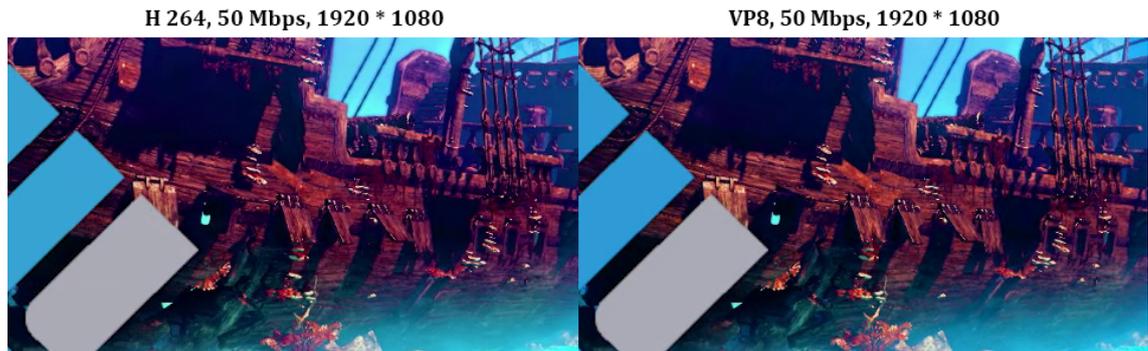


Figure 16. Quality comparison of the codecs from the trailer. (Adobe 2015)

VP8 produces slightly more vibrant colours, whereas H.264 has a bit better colour reproduction in the blacks. For an average viewer, there is basically no difference between the two. On the technical side, both codecs performed equally well (see figures 17 and 18) There were only a few lost frames in both of them.

Current media / stream statistics	
▲ Audio	
Decoded	7798 blocks
Played	7798 buffers
Lost	0 buffers
▲ Video	
Decoded	5331 blocks
Displayed	5306 frames
Lost	17 frames
▲ Input/Read	
Media data size	472354 KiB
▶ Input bitrate	1273 kb/s
Demuxed data size	472252 KiB
Content bitrate	1577 kb/s
Discarded (corrupted)	0
Dropped (discontinued)	0
▲ Output/Written/Sent	
Sent	0 packets
Sent	0 KiB
Upstream rate	0 kb/s

Figure 17. Data statistics of video encoded with VP8, played in VLC player. (VideoLAN Organisation 2015)

VP8 lost only 17 frames out of over 5300 frames and there were no corrupted data in the output file. H.264 had slightly worse statistics, as it lost 30 frames out of over 5300. However, both the numbers are so small that with the 60 frames per second rate, there was not a single visible fault in either of the codecs.

Current media / stream statistics	
▲ Audio	
Decoded	4199 blocks
Played	4199 buffers
Lost	0 buffers
▲ Video	
Decoded	5367 blocks
Displayed	5308 frames
Lost	30 frames
▲ Input/Read	
Media data size	509966 KiB
▶ Input bitrate	13796 kb/s
Demuxed data size	506609 KiB
Content bitrate	13702 kb/s
Discarded (corrupted)	0
Dropped (discontinued)	0
▲ Output/Written/Sent	
Sent	0 packets
Sent	0 KiB
Upstream rate	0 kb/s

Figure 18. Data statistics of the video encoded with H.264 codec, (VideoLAN Organisation 2015)

Because the file size is nearly the same, with a difference of about 10 %, it cannot be found a reason why not support the free, open source codec instead of the commercial one.

## 7 Future Perspectives

When technology runs forward, this usually means larger screens in video production, but first and foremost, sharper images and furthermore bigger resolutions. While H.264 and VP8 produce excellent quality and small sized files, problems are occurring when the video screen and resolutions grow larger and larger.

New generation of of video codecs are being developed to meet the needs. Continuing with the codecs described in this thesis, the next generations are H.265/HEVC (High Efficiency Video Codec) from the commercially patented side, and VP9, from the open source side. In fact, H.265 has already been approved for a standard (ITU 2013), but the codec is not yet fully optimized for a user to utilize. As seen from figure 19, H.265 should support resolutions up to 8K, which is 7680 pixels wide and 4320 pixels high. (Sullivan 2012, 1665.)

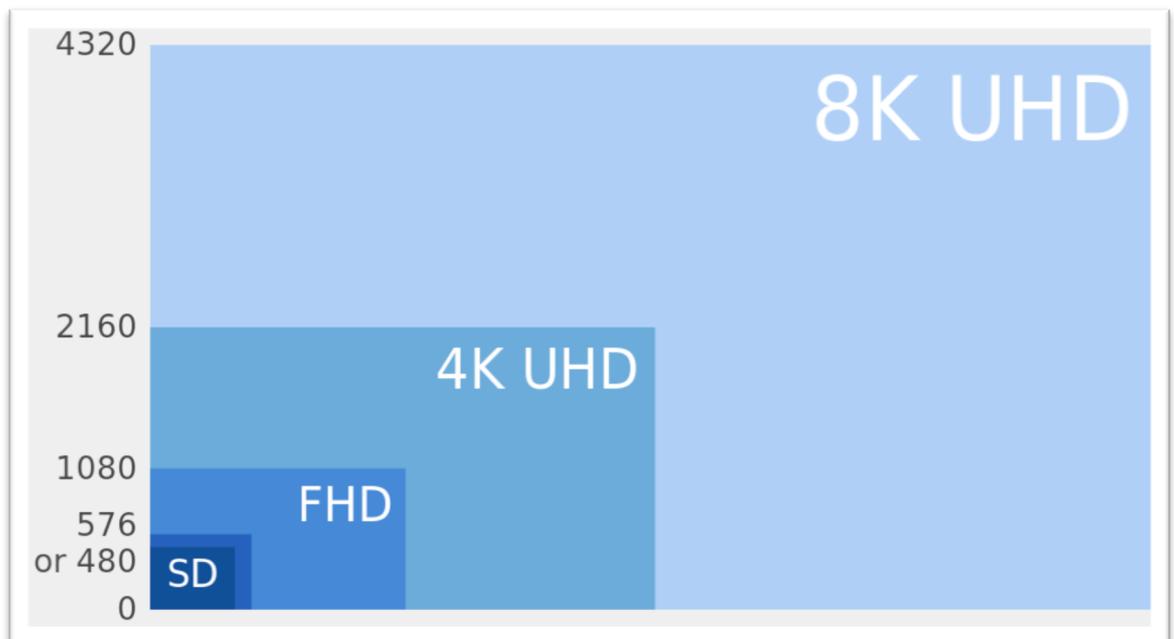


Figure 19. The size of 8K resolution compared to smaller resolutions. (Wikimedia Commons 2015)

As discussed in chapter 2, these kinds of resolutions are out of reach when it comes to the older codecs, because the bit rates grow too large for web streams or other medias to handle. This is also the reason why new techniques are tested regularly, like in this final year project, where VP9 was tried to implement as main codec.

## 8 Conclusions

The purpose of this thesis was to explain the basics of open source codecs and how and why they are used and why they should be used more and more in online video publishing. The first chapters handled the basics and the history of video and open source, followed by chapters about codecs and their usage in video production. The last chapters describe what was done in the final year project.

The study came to a conclusion that H.264/AVC/MPEG-4 Part 10 codec is not the standard for video without a reason. It produces magnificent quality video with a great overall ratio between size and quality. H.264 is a follow up for MPEG-1 and MPEG-2, and has been around for over a decade. However, during the past decade, the world of information technology has evolved towards a new, more open atmosphere and free software, which is why encouraging and supporting VP8 over H.264 was chosen.

VP8 and WebM are designed specifically for online web use from the start. (WebM 2015) Whereas H.264 and most of the other codecs are developed for professional use, so that they could also be used in film production and such, VP8 is meant for anyone to use and publish video.

For online video, which was the main topic of this thesis, it is crucially important that the video is easy to play and rewind, and that it can easily be played on various different platforms and devices. On the one hand, the web can be browsed with small, three inch sized mobile phones, but on the other end with enormous, metres long screens. This means that the resolution has got to be responsive throughout from tiny to full HD and even larger. Most importantly, it needs to make optimal use of the network. The thesis came to the conclusion that VP8 is overall a better codec for today's web use. Whereas the web has always been an open platform, it also needs an open video format, and this is why the VP8 was developed, and this is also the reason for writing this thesis.

Open source software in general promotes and encourages people to think by themselves, leads to new innovations when anyone can participate, and for the most important part it *is free* for anyone to use, in any production, anywhere in the world. There are absolutely no reasons for an amateur film editor or an amateur producer to support

commercial, corporate patented and owned methods, and pay for them, when an equally good or better system is available without costs.

While corporations are fighting hard for their patents, great, community supportive companies like Google, Mozilla, and Xiph Open Source Community are developing open source software with supportive inspired people. Requiring money from small business entrepreneurs for high quality video makes it harder for publishers and entrepreneurs to produce more quality content and services. Joint Video Group, Moving Picture Experts Group and others have done remarkable work with video compression and companies like theirs should not be look down on. However, monetizing everything, keeping inventions in secret, preventing other, ingenious and creative people to enrich the product should not be the industry standard.

Anyone should be able to produce a great looking video and have it online to show all around the world.

## References

Adobe. Adobe Premiere Pro CC [computer program]. Version 8.2.0. 2014.2 Release. San Jose: California. Adobe Systems; 2015.

Adobe. Press Release [online]. Adobe Systems. San Jose: California;  
URL: <http://www.adobe.com/news-room/pressreleases/201412/121114Q4FY2014results.html?PID=2159997>. Accessed 26 April 2015 .

Bolles Brendan. GitHub. WebM plug-ins for Adobe programs. [online]: 2015.  
URL: <https://github.com/fnordware/AdobeWebM> Accessed 26 April 2015

Burg, Jennifer. The Science of Digital Media; Upper Saddle River: Pearson Education Inc. 2009.

Canon. Canon EOS C100. [online] Cinema EOS Cameras. 2015.  
URL: [http://www.canon.fi/for\\_home/product\\_finder/digital\\_cinema/cinema\\_eos\\_cameras/eos\\_c100/](http://www.canon.fi/for_home/product_finder/digital_cinema/cinema_eos_cameras/eos_c100/). Accessed 26 April 2015.

Dhanani Suhel, Parkel Michael. Digital Video Processing for Engineers. Oxford: Elsevier Inc; 2013.

Fischer Walter. Digital Video and Audio Broadcasting Technology. Third Edition. London. Springer. 2010

Frozenbyte Inc. Trine3.com [online] Finland: Helsinki; 2015.  
URL: [www.trine3.com](http://www.trine3.com). Accessed 26 April 2015.

IDC Corporate USA [online]. Framingham; United States, Maine; 2015  
URL: <http://www.idc.com/prodserv/smartphone-os-market-share.jsp>. Accessed 26 April 2015.

International Organisation for Standardisation (ISO). Switzerland; 2015  
URL: <http://www.iso.org/iso/home/standards.htm>. Accessed 26 April 2015.

International Telecommunication Union (ITU), United Nations; 2015  
URL: <http://www.itu.int/ITU-T/recommendations/rec.aspx?rec=11885>. Accessed 26 April 2015.

Lakhani Karim, von Hippel Eric. How Open Source Software Works: "Free" User-To-User Assistance. [online]. Cambridge: Elsevier; 2002  
URL: [http://www.ee.oulu.fi/~vassilis/courses/socialweb10F/reading\\_material/2/lakhani00-HowOpenSourceSoftwareWorks.pdf](http://www.ee.oulu.fi/~vassilis/courses/socialweb10F/reading_material/2/lakhani00-HowOpenSourceSoftwareWorks.pdf). Accessed 26 April 2015.

Microsoft Corporation. Windows 8. [computer program] Version 6.3. Washington: Redmont; 2015.

MPEG-LA .AVC / H.264 Licensors. [online]. 2015  
URL: <http://www.mpegla.com/main/programs/AVC/Pages/Licensors.aspx>. Accessed 26 April 2015.

Netmarketshare. Operating System Market Share. [online]. 2015  
URL: <https://www.netmarketshare.com/operating-system-market-share.aspx?qprid=10&qpcustomd=1>. Accessed 26 April 2015.

Ookla.. Net index. [online]. Ookla Industries. 2015  
URL: <http://www.netindex.com/download/map>. Accessed 26 April 2015.

Open264. Home page. [online]. 2015  
URL: <http://www.openH.264.org/index.html>. Accessed 26 April 2015

Oxford Dictionaries. Definition of Video. Oxford University Press. 2015. [online].  
URL: <http://www.oxforddictionaries.com/definition/english/video?searchDictCode=all> .  
Accessed 26 April 2015.

Richardson, Iain. The H.264 Advanced Video Compression Standard. Second edition. United Kingdom: John Wiley & Sons Ltd; 2010.

Samsung. Specifications for Samsung Galaxy S3 phone. [online]. Samsung; 2015  
URL: <http://www.samsung.com/global/galaxys3/specifications.html>. Accessed 26 April 2015.

Stump, David. Digital Cinematography: fundamentals, tools, techniques, and workflows. Burlington: Taylor & Francis; 2014.

Sullivan, Gary J. Overview of the High Efficiency. [online]. IEEE Transactions on circuits and systems for video technology, VOL 22, No. 12; 2012.  
URL: <http://ieeexplore.ieee.org/stamp/stamp.jsp?reload=true&tp=&arnumber=6316136>.  
Accessed 26 April 2015.

Symes Peter. Digital Video Compression. New York: McGraw-Hill; 2004.

Theora. Xiph.Org. [online]. 2015  
URL: <http://www.theora.org/>. Accessed 26 April 2015.

VidéoLAN Organisation. VLC [computer program]. Version 2.2.1. France: Paris; 2015.

WebM. The WebM Project. [online]. 2015  
URL: <http://www.WebMproject.org/about/>. Accessed 26 April 2015.

Wikimedia Commons. 8K UHD picture. [online]. 2015

URL: [http://upload.wikimedia.org/wikipedia/commons/thumb/2/20/8K\\_UHD%2C\\_4K\\_SHD%2C\\_FHD\\_and\\_SD.svg](http://upload.wikimedia.org/wikipedia/commons/thumb/2/20/8K_UHD%2C_4K_SHD%2C_FHD_and_SD.svg). Accessed 26 April 2015.

W3Tech. Usage of Operating systems online. [online]. 2015

URL: [http://w3techs.com/technologies/overview/operating\\_system/all](http://w3techs.com/technologies/overview/operating_system/all). Accessed 26 April 2015.

W3Schools. Browser statistics and Trends. [online]. 2015

URL: [http://www.w3schools.com/browsers/browsers\\_stats.asp](http://www.w3schools.com/browsers/browsers_stats.asp). Accessed 26 April 2015.

Youtube. Youtuben käyttöehdot. [online] 2015

URL: <https://www.youtube.com/static?template=terms> Accessed 26 April 2015.

YouTube. YouTube Statistics. [online]. 2015

URL: <https://www.youtube.com/yt/press/fi/statistics.html>. Accessed 26 April 2015.

YouTube. YouTube-logo. [online] 2015

URL: <http://www.youtube.com/yt/brand/fi/downloads.html>. Accessed 26 April 2015.

YouTubeCreator. The Official YouTube Partners and Creators blog. [online]. 2014

URL: <http://youtubecreator.blogspot.ie/2014/06/look-ahead-creator-features-coming-to.html>. Accessed 26 April 2015.