Chapter No. 1
"An Introduction to Web-based Real-Time Communication"
In this package, you will find:

A Biography of the author of the book
A preview chapter from the book, Chapter NO.1 "An Introduction to Web-based Real-Time Communication"
A synopsis of the book’s content
Information on where to buy this book

About the Author

Rob Manson is the CEO and co-founder of buildAR.com, the world's leading Augmented Reality Content Management System. Rob is the Chairman of the W3C Augmented Web Community Group, and an Invited Expert with the ISO, W3C, and the Khronos Group. He is one of the co-founders of ARStandards.org and is an active evangelist within the global AR and standards communities. He is regularly invited to speak on the topics of the Augmented Web, Augmented Reality, WebRTC, and multi-device platforms.

I'd like to thank Alex, my wife and business partner—yes that's as crazy as it sounds! She's a great inspiration and always happy to put up with my creative ideas for using new technologies. She makes both my ideas and me as a person better in every way. I'd also like to thank Maggie and Todd for providing feedback and working with me on all our Multi-Device, WebRTC, and Augmented Web projects. I'm constantly amazed by just how much our team can achieve and you guys are the backbone that make this happen. I'm proud to say I work with you both.

For More Information:
Getting Started with WebRTC

*Getting Started with WebRTC* provides all the practical information you need to quickly understand what WebRTC is, how it works, and how you can add it to your own web applications. It includes clear working examples designed to help you get started with building WebRTC-enabled applications right away.

WebRTC delivers Web-based Real-Time Communication, and it is set to revolutionize our view of what the "Web" really is. The ability to stream audio and video from browser to browser alone is a significant innovation that will have far reaching implications for the telephony and video conferencing industries. But this is just the start. Opening raw access to the camera and microphone for JavaScript developers is already creating a whole new dynamic web that allows applications to interact with users through voice, gesture, and all kinds of new options.

On top of that, WebRTC also introduces real-time data channels that will allow interaction with dynamic data feeds from sensors and other devices. This really is a great time to be a web developer! However, WebRTC can also be quite daunting to get started with and many of its concepts can be new or a little confusing for even the most experienced web developers.

It's also important to understand that WebRTC is not really a single technology, but more a collection of standards and protocols, and it is still undergoing active evolution. The examples covered in this book are based on the latest version of the pre-1.0 version of the WebRTC standards at the time of writing. However, there are some areas of these standards that are under active debates and may change over the next year. The first is the way that the Session Description Protocol is integrated into the WebRTC call management process. The second is the general use of the overall offer/answer model that underlies the call setup process. And finally, there is also a strong push for the WebRTC standards to integrate the new Promise (previously known as Futures) design pattern. This all shows that this is a cutting edge, active, and exciting technology area, and that now is a great time to get involved as it grows and evolves.

We hope you appreciate this practical guide and that it makes it easy for you to get started with adding WebRTC to your applications right away.

For More Information:
What This Book Covers

Chapter 1, An Introduction to Web-based Real-Time Communication, introduces you to the concepts behind the new Web-based Real-Time Communication (WebRTC) standards.

Chapter 2, A More Technical Introduction to Web-based Real-Time Communication, takes you to the technical concepts behind the new Web-based Real-Time Communication (WebRTC) standards.

Chapter 3, Creating a Real-time Video Call, shows you how to use the MediaStream and RTCPeerConnection APIs to create a working peer-to-peer video chat application between two people.

Chapter 4, Creating an Audio Only Call, teaches you how to turn the video chat application we developed in the previous chapter into an audio only call application.

Chapter 5, Adding Text-based Chat, explains how to extend the video chat application we developed in Chapter 2, A More Technical Introduction to Web-based Real-Time Communication, to add support for text-based chat between the two users.

Chapter 6, Adding File Sharing, deals with how to extend the video chat application we developed in Chapter 2, A More Technical Introduction to Web-based Real-Time Communication and Chapter 4, Creating an Audio Only Call, to add support for file sharing between the two users.

Chapter 7, Example Application 1 — Education and E-learning, maps out what is involved in introducing WebRTC into e-learning applications.

Chapter 8, Example Application 2 — Team Communication, shows what is involved in introducing WebRTC into team your communication applications.

For More Information:
An Introduction to Web-based Real-Time Communication

This chapter introduces you to the concepts behind the new Web-based Real-Time Communication (WebRTC) standards. After reading this chapter, you will have a clear understanding of:

- What is WebRTC
- How you can use it
- Which web browsers support it

Introducing WebRTC

When the World Wide Web (WWW) was first created in the early 1990's, it was built upon a page-centric model that used HREF-based hyperlinks. In this early model of the web, browsers navigated from one page to another in order to present new content and to update their HTML-based user interfaces.

Around the year 2000, a new approach to web browsing had started to develop, and by the middle of that decade, it had become standardized as the XMLHttpRequest (XHR) API. This new XHR API enabled web developers to create web applications that didn't need to navigate to a new page to update their content or user interface. It allowed them to utilize server-based web services that provided access to structured data and snippets of pages or other content. This led to a whole new approach to the web, which is now commonly referred to as Web 2.0. The introduction of this new XHR API enabled services such as Gmail, Facebook, Twitter, and more to create a much more dynamic and social web for us.

For More Information:
Now the web is undergoing yet another transformation that enables individual web browsers to stream data directly to each other without the need for sending it via intermediary servers. This new form of peer-to-peer communication is built upon a new set of APIs that is being standardized by the Web Real-Time Communications Working Group available at [http://www.w3.org/2011/04/webrtc/](http://www.w3.org/2011/04/webrtc/) of the World Wide Web Consortium (W3C), and a set of protocols standardized by Real-Time Communication in WEB-browsers Working Group available at [http://tools.ietf.org/wg/rtcweb/](http://tools.ietf.org/wg/rtcweb/) of the Internet Engineering Task Force (IETF).

Just as the introduction of the XHR API led to the Web 2.0 revolution, the introduction of the new WebRTC standards is creating a new revolution too.

It's time to say hello to the real-time web!

**Uses for WebRTC**

The real-time web allows you to set up dynamic connections to other web browsers and web-enabled devices quickly and easily. This opens the door to a whole new range of peer-to-peer communication, including text-based chat, file sharing, screen sharing, gaming, sensor data feeds, audio calls, video chat, and more. You can now see that the implications of WebRTC are very broad. Direct and secure peer-to-peer communication between browsers will have a big impact on the modern web, reshaping the way we use the physical networks that make up the Internet.

Direct peer-to-peer connections often provide lower latency, making gaming, video streaming, sensor data feeds, and so on, appear faster and more interactive or real-time, hence the use of this term.

Secure peer-to-peer connections allow you to exchange information privately without it being logged or managed by intermediary servers. This reduces the need for some large service providers while creating opportunities for people to create new types of services and applications. It introduces improved privacy for some individuals while it may also create new complexities for regulators and law enforcement organizations.

And the efficient peer-to-peer exchange of binary data streams removes the need to serialize, re-encode, or convert this data at each step in the process. This leads to a much more efficient use of network and application resources, as well as creating a less error prone and more robust data exchange pipeline.

This is just a brief overview of how you can use WebRTC, and by the end of this book, you will have all the information you need to start turning your own new ideas into practical applications.

For More Information:

Try WebRTC yourself right now!

The goal of this book is to get you started with WebRTC, so let's do that right now. You can easily find out if your browser supports the camera access functionality by visiting one of the existing demo sites such as http://www.simpl.info/getusermedia, and if it does, you should be prompted to provide permission to share your camera. Once you provide this permission, you should see a web page with a live video stream from your PC or mobile devices' video camera, and be experiencing the interesting sensation of looking at a video of yourself staring right back at you. That's how simple it is to start using WebRTC.

Now, perhaps you'd like to try using it to communicate with another person. You can do this by visiting another demo site such as http://apprtc.appspot.com, which will create a unique URL for your video chat. Just send this URL to another person with a browser that also supports WebRTC, and once they open that page, you should see two video elements displayed on the page: one from your local video camera and one from the other person's video camera. There's a lot of complex negotiation that's gone on in the background, but assuming your browser supports WebRTC and your network doesn't actively prevent it, then you should now have a clear idea of just how easy it is to use.

But what web browsers support WebRTC? Let's find out.

Browser compatibility

The WebRTC standards landscape is home to one of the fastest evolving communities on the web. One of the biggest challenges this creates is that of compatibility and interoperability. Here is an overview of what this is up to today and how to stay up-to-date as this continues to evolve.

Chrome and Firefox on the PC

At the time this chapter was written, WebRTC was supported as default by Chrome and Firefox on mainstream PC Operating Systems such as Mac OS X, Windows, and Linux. And most importantly, these two key implementations have been shown to communicate well with each other through a range of interoperability tests.

Have a look at the Hello Chrome, it's Firefox calling! blog post at https://hacks.mozilla.org/2013/02/hello-chrome-its-firefox-calling/.
Chrome and Firefox on Android

WebRTC is also available for Chrome and Firefox on the Android platform; however, currently you must manually configure certain settings to enable this functionality.

Here are the key steps you need to enable this for Chrome. These are from the Chrome for Android release notes posted on the discuss-webrtc forum available at https://groups.google.com/forum/#!topic/discuss-webrtc/uFOMhd-AG0A:

To enable WebRTC on Chrome for Android:

1. Type in `chrome://flags/` in the omnibox to access the flags.
2. Scroll about a third down and enable the **Enable WebRTC** flag.
3. You will be asked to relaunch the browser at the bottom of the page in order for the flag to take effect.

For More Information:

Here are the key steps you need to enable WebRTC for Firefox. These are from a post on the Mozilla Hacks blog about the new Firefox for Android release available at https://hacks.mozilla.org/2013/04/webrtc-update-our-first-implementation-will-be-in-release-soon-welcome-to-the-party-but-please-watch-your-head/:

You can enable it by setting both the `media.navigator.enabled` pref and the `media.peerconnection.enabled` pref to ”true” (browse to about:config and search for `media.navigator.enabled` and `media.peerconnection.enabled` in the list of prefs).

![Enabling WebRTC using Firefox settings on Android](image)

**Opera**

Opera has been an active supporter of the WebRTC movement and has implemented early versions of this standard in previous releases of their browsers. But at the time this chapter was written, they were working to port their collection of browsers to the WebKit platform based on the open Chromium project. So, until this migration activity is complete, their support for WebRTC is currently listed as unavailable.
However, since the Chromium project is closely related to Chrome, which is also built upon the WebKit platform, it is expected that Opera's support for WebRTC will develop quickly after this migration is complete.

**Microsoft**


As yet, it has not announced any timeline as to when Internet Explorer may support WebRTC, but it is currently possible to use WebRTC within Internet Explorer using the Chrome Frame solution available at [https://developers.google.com/chrome/chrome-frame/](https://developers.google.com/chrome/chrome-frame/).

Microsoft has also recently released prototypes that show interoperability in the form of a voice chat application connecting Chrome on a Mac and IE10 on Windows available at [http://blogs.msdn.com/b/interoperability/archive/2013/01/17/ms-open-tech-publishes-html5-labs-prototype-of-a-customizable-ubiquitous-real-time-communication-over-the-web-api-proposal.aspx](http://blogs.msdn.com/b/interoperability/archive/2013/01/17/ms-open-tech-publishes-html5-labs-prototype-of-a-customizable-ubiquitous-real-time-communication-over-the-web-api-proposal.aspx). This shows that one way or another, Microsoft understands the significance of the WebRTC movement, and it is actively engaging in the standards discussions.

**Apple**

Apple has not yet made any announcement about when they plan to support WebRTC in Safari on either OS X or iOS. So far, the only application that has made WebRTC available on iOS is an early proof of concept browser created by Ericsson Labs named **Bowser**, and is available at [http://labs.ericsson.com/apps/bowser](http://labs.ericsson.com/apps/bowser).

Bowser is based upon a very early experimental version of the WebRTC standards, and it does not interoperate with any of the other mainstream web browsers.

However, as Safari is also based upon the WebKit platform just like Chrome and Opera, there should be no major technical barriers to prevent Apple from enabling WebRTC on both their mobile and PC browsers.

For More Information:
Staying up-to-date
It is also important to note that WebRTC is not a single API, but really a collection of APIs and protocols defined by a variety of Working Groups, and that the support for each of these are developing at different rates on different browsers and operating systems.

A great way to see where the latest level of support has reached is through services such as http://caniuse.com, which tracks broad adoption of modern APIs across multiple browsers and operating systems.

And, you should also check out the open project at http://www.webrtc.org, which is supported by Google, Mozilla, and Opera. This project provides a set of C++ libraries that are designed to help browser and application developers quickly and easily implement standards compliant with WebRTC functionality. It is also a useful site to find the latest information on browser support and some great WebRTC demos.

Summary
You should now have a clear overview of what the term WebRTC means and for what it can be used. You should be able to identify which browsers support WebRTC and have all the resources you need to find the latest up-to-date information on how this is evolving. You should also have been able to try the different aspects of WebRTC for yourself quickly and easily using your own browser if you so choose.

Next, we will take a more technical look at how the different WebRTC API components all fit together.

Then, we will start by fleshing out the simple peer-to-peer video call scenario into a fully working application.

Later, we will explore how this can be simplified down to just an audio only call or extended with text-based chat and file sharing.

And then, we will explore two real-world application scenarios based upon e-learning and team communication.

For More Information:
Where to buy this book


Free shipping to the US, UK, Europe and selected Asian countries. For more information, please read our shipping policy.

Alternatively, you can buy the book from Amazon, BN.com, Computer Manuals and most internet book retailers.