Cardboard is Virtual Reality for Everyone

Jonathan Linowes

April 2016

In this article, by **Jonathan Linowes** and **Matt Schoen**, authors of the book Cardboard VR Projects for Android, introduce and define Google Cardboard.

(For more resources related to this topic, see here.)

Welcome to the exciting new world of virtual reality! We're sure that, as an Android developer, you want to jump right in and start building cool stuff that can be viewed using Google Cardboard. Your users can then just slip their smartphone into a viewer and step into your virtual creations. Let's take an outside-in tour of VR, Google Cardboard, and its Android SDK to see how they all fit together.

Why is it called Cardboard?

It all started in early 2014 when Google employees, David Coz and Damien Henry, in their spare time, built a simple and cheap stereoscopic viewer for their Android smartphone. They designed a device that can be constructed from ordinary cardboard, plus a couple of lenses for your eyes, and a mechanism to trigger a button "click."

The viewer is literally made from cardboard. They wrote software that renders a 3D scene with a split screen, one view for the left eye, and another view, with offset, for the right eye. Peering through the device, you get a real sense of 3D immersion into the computer generated scene. It worked! The project was then proposed and approved as a "20% project" (where employees may dedicate one day a week for innovations), funded, and joined by other employees.

In fact, Cardboard worked so well that Google decided to go forward, taking the project to the next level and releasing it to the public a few months later at Google I/O 2014.

Since its inception, Google Cardboard has been accessible to hackers, hobbyists, and professional developers alike. Google open sourced the viewer design for anyone to download the schematics and make their own, from a pizza box or from whatever they had lying around. One can even go into business selling precut kits directly to consumers. An assembled Cardboard viewer is shown in the following image:

The Cardboard project also includes a **software development kit** (**SDK**) that makes it easy to build VR apps. Google has released continuous improvements to the software, including both a native Java SDK as well as a plugin for the Unity 3D game engine (https://unity3d.com/).

Since the release of Cardboard, a huge number of applications have been developed and made available on the Google Play Store. At Google I/O 2015, Version 2.0 introduced an upgraded design, improved software, and support for Apple iOS.

Google Cardboard has rapidly evolved in the eye of the market from an almost laughable toy into a serious new media device for certain types of 3D content and VR experiences. Google's own Cardboard demo app has been downloaded millions of times from the Google Play store. The New York Times distributed about a million cardboard viewers with its November 8, Sunday issue back in 2015.

Cardboard is useful for viewing 360-degree photos and playing low-fidelity 3D VR games. It is universally accessible to almost anyone because it runs on any Android or iOS smartphone.

For developers who are integrating 3D VR content directly into Android apps, Google Cardboard is a way of experiencing virtual reality that is here to stay.

A gateway to VR

Even in a very short time, it's been available; this generation of consumer virtual reality, whether Cardboard or Oculus Rift, has demonstrated itself to be instantly compelling, immersive, entertaining, and "a game changer" for just about everyone who tries it. Google Cardboard is especially easy to access with a very low barrier to use. All you need is a smartphone, a low-cost Cardboard viewer (as low as $5 USD), and free apps downloaded from Google Play (or Apple App Store for iOS).

Google Cardboard has been called a **gateway to VR**, perhaps in reference to marijuana as a "gateway drug" to more dangerous illicit drug abuse? We can play with this analogy for a moment, however, decadent. Perhaps Cardboard will give you a small taste of VR's potential.

You'll want more. And then more again. This will help you fulfill your desire for better, faster, more intense, and immersive virtual experiences that can only be found in higher end VR devices. At this point, perhaps there'll be no turning back, you're addicted!

Yet as a Rift user, I still also enjoy Cardboard. It's quick. It's easy. It's fun. And it really does work, provided I run apps that are appropriately designed for the device.

I brought a Cardboard viewer in my backpack when visiting my family for the holidays. Everyone enjoyed it a lot. Many of my relatives didn't even get past the standard Google Cardboard demo app, especially its 360-degree photo viewer. That was engaging enough to entertain them for a while. Others jumped to a game or two or more. They wanted to keep playing and try new experiences. Perhaps it's just the novelty. Or, perhaps it's the nature of this new medium. The point is that Google Cardboard provides an immersive experience that's enjoyable, useful, and very easily accessible. In short, it is amazing.

Then, show them an HTC Vive or Oculus Rift. Holy Cow! That's really REALLY amazing! We're not here to talk about the higher end VR devices, except to contrast it with Cardboard and to keep things in perspective.

Once you try desktop VR, is it hard to "go back" to mobile VR? Some folks say so. But that's almost silly. The fact is that they're really separate things.

As discussed earlier, desktop VR comes with much higher processing power and other high-fidelity features, whereas mobile VR is limited to your smartphone. If one were to try and directly port a desktop VR app to a mobile device, there's a good chance that you'll be disappointed.

It's best to think of each as a separate media. Just like a desktop application or a console game is different from, but similar to, a mobile one. The design criteria may be similar but different. The technologies are similar but different. The user expectations are similar but different. Mobile VR may be similar to desktop VR, but it's different.

To emphasize how different Cardboard is from desktop VR devices, it's worth pointing out that Google has written their manufacturer's specifications and guidelines.

"Do not include a headstrap with your viewer. When the user holds the Cardboard with their hands against the face, their head rotation speed is limited by the torso rotational speed (which is much slower than the neck rotational speed). This reduces the chance of "VR sickness" caused by rendering/IMU latency and increases the immersiveness in VR."

The implication is that Cardboard apps should be designed for shorter, simpler, and somewhat stationary experiences.

Let's now consider the other ways where Cardboard is a gateway to VR.

We predict that Android will continue to grow as a primary platform for virtual reality into the future. More and more technologies will get crammed into smartphones. And this technology will include features specifically with keeping VR in mind:

- Faster processors and mobile GPUs
- Higher resolution screens
- Higher precision motion sensors
- Optimized graphics pipelines
- Better software
- Tons of more VR apps

Mobile VR will not give way to desktop VR; it may even eventually replace it.

Furthermore, maybe soon we'll see dedicated mobile VR headsets that have the guts of a smartphone built-in without the cost of a wireless communications' contract. No need to use your own phone. No more getting interrupted while in VR by an incoming call or notification. No more rationing battery life in case you need to receive an important call or otherwise use your phone. These dedicated VR devices will likely be Android-based.

The value of low-end VR

Meanwhile, Android and Google Cardboard are here today, on our phones, in our pockets, in our homes, at the office, and even in our schools.

Google Expeditions, for example, is Google's educational program for Cardboard (https://www.google.com/edu/expeditions/), which allows K-12 school children to take virtual field trips to "places a school bus can't," as they say, "around the globe, on the surface of Mars, on a dive to coral reefs, or back in time." The kits include Cardboard viewers and Android phones for each child in a classroom, plus an Android tablet for the teacher. They're connected with a network. The teacher can then guide students on virtual field trips, provide enhanced content, and create learning experiences that go way beyond a textbook or classroom video, as shown in the following image:

Expeditions: Take your students to places a school bus can't

The entire Internet can be considered a world-wide publishing and media distribution network. It's a web of hyperlinked pages, text, images, music, video, JSON data, web services, and many more. It's also teeming with 360-degree photos and videos. There's also an ever growing amount of three-dimensional content and virtual worlds. Would you consider writing an Android app today that doesn't display images? Probably not. There's a good chance that your app also needs to support sound files, videos, or other media. So, pay attention. Three-dimensional Cardboard-enabled content is coming quickly. You might be interested in reading this article now because VR looks fun. But soon enough, it may be a customer-driven requirement for your next app.

Some examples of types of popular Cardboard apps include:

- **360 degree photo viewing**, for example, Google's Cardboard demo (https://play.google.com/store/apps/details?id=com.google.samples.apps.ca...) and Cardboard Camera (https://play.google.com/store/apps/details?id=com.google.vr.cyclops)
- **Video and cinema viewing**, for example, a Cardboard theatre (https://play.google.com/store/apps/details?id=it.couchgames.apps.cardboa...)
- **Roller coasters and thrill rides**, for example, VR Roller Coaster (https://play.google.com/store/apps/details?id=com.frag.vrrollercoaster)
- **Cartoonish 3D games**, for example, Lamber VR (https://play.google.com/store/apps/details?id=com.archiactinteractive.Lf...)
- **First person shooter games**, for example, Battle 360 VR (https://play.google.com/store/apps/details?id=com.oddknot.battle360vr)
- **Creepy scary stuff**, for example, Sisters (https://play.google.com/store/apps/details?id=com.otherworld.Sisters)
- **Educational experiences**, for example, Titans of Space (https://play.google.com/store/apps/details?id=com.drashvr.titansofspacec...)
- **Marketing experiences**, for example, Volvo Reality (https://play.google.com/store/apps/details?id=com.volvo.volvoreality)

And much more. Thousands more. The most popular ones have had hundreds of thousands of downloads (the Cardboard demo app itself has millions of downloads).

Cardware!

Let's take a look at the variety of Cardboard devices that are available. There's a lot of variety.

Obviously, the original Google design is actually made from cardboard. And manufacturers have followed suit, offering cardboard Cardboards directly to consumers—brands such as Unofficial Cardboard, DODOCase, and IAmCardboard were among the first.

Google provides the specifications and schematics free of charge (refer to https://www.google.com/get/cardboard/manufacturers/). The basic viewer design consists of an enclosure body, two lenses, and an input mechanism. The *Works with Google Cardboard* certification program indicates that a given viewer product meets the Google standards and works well with Cardboard apps.

The viewer enclosure may be constructed from any material: cardboard, plastic, foam, aluminum, and so on. It should be lightweight and do a pretty good job of blocking the ambient light.

The lenses (I/O 2015 Edition) are 34 mm diameter aspherical single lenses with an 80 degree circular FOV (field of view) and other specified parameters.

The input trigger ("clicker") can be one of the several alternative mechanisms. The simplest is none, where the user must touch the smartphone screen directly with their finger to trigger a click. This may be inconvenient since the phone is sitting inside the viewer enclosure but it works. Plenty of viewers just include a hole to stick your finger inside. Alternatively, the original Cardboard utilized a small ring magnet attached to the outside of the viewer. The user can slide the magnet and this movement is sensed by the phone's magnetometer and recognized by the software as a "click". This design is not always reliable because the location of the magnetometer varies among phones. Also, using this method, it is harder to detect a "press and hold" interaction, which means that there is only one type of user input "event" to use within your application.

Lastly, Version 2.0 introduced a button input constructed from a conductive "strip" and "pillow" glued to a Cardboard-based "hammer". When the button is pressed, the user's body charge is transferred onto the smartphone screen, as if he'd directly touched the screen with his finger. This clever solution avoids the unreliable magnetometer solution, instead uses the phone's native touchscreen input, albeit indirectly.

It is also worth mentioning at this point that since your smartphone supports Bluetooth, it's possible to use a handheld Bluetooth controller with your Cardboard apps. This is not part of the Cardboard specifications and requires some extra configuration; the use of a third-party input handler or controller support built into the app. A mini Bluetooth controller is shown in the following image:

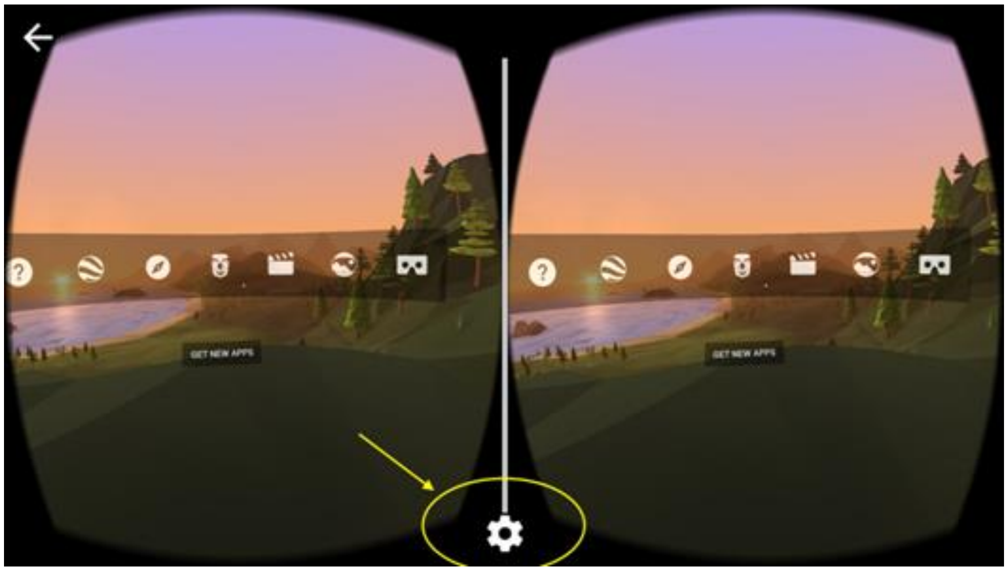
Cardboard viewers are not necessarily made out of cardboard. Plastic viewers can get relatively costly. While they are more sturdy than cardboards, they fundamentally have the same design (assembled). Some devices include adjustable lenses, for the distance of the lenses from the screen, and/or the distance between your eyes (IPD or inter-pupillary distance). The Zeiss VR One, Homido, and Sunnypeak devices were among the first to become popular.

Some manufacturers have gone out of the box (pun intended) with innovations that are not necessarily compliant with Google's specifications but provide capabilities beyond the Cardboard design. A notable example is the Wearality viewer (http://www.wearality.com/), which includes an exclusive patent 150-degree **field of view** (**FOV**) double Fresnel lens. It's so portable that it folds up like a pair of sunglasses. The Wearality viewer is shown in the following image:

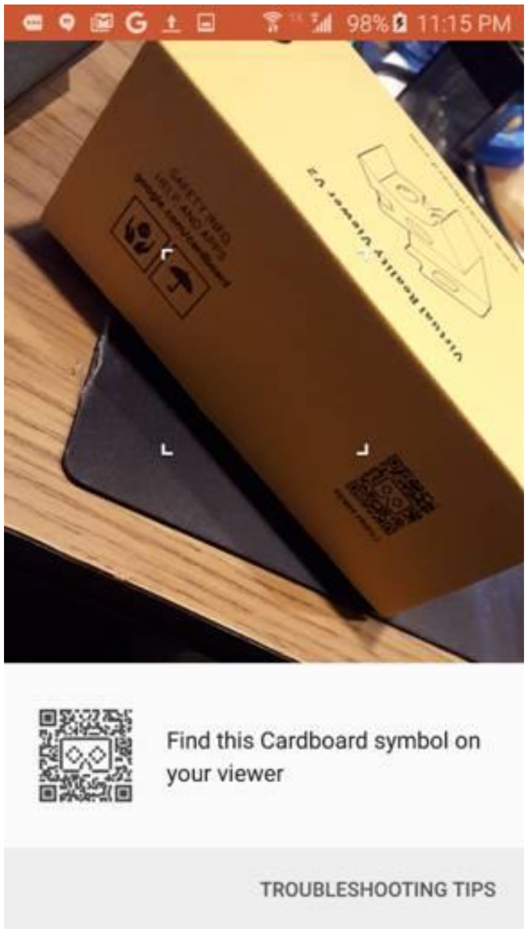
Configuring your Cardboard viewer

With such a variety of Cardboard devices and variations in lens distance, field of view, distortion, and so on, Cardboard apps must be configured to a specific device's attributes. Google provides a solution to this as well. Each Cardboard viewer comes with a unique QR code and/or NFC chip, which you scan to configure the software for that device. If you're interested in calibrating your own device or customizing your parameters, check out the profile generator tools at https://www.google.com/get/cardboard/viewerprofilegenerator/.

To configure your phone to a specific Cardboard viewer, open the standard Google Cardboard app, and select the **Settings** icon in the center bottom section of the screen, as shown in the following image:



Then, point the camera to the QR code for your particular Cardboard viewer:


98% 11:15 PM
Virtual Reality Viewer V2
Find this Cardboard symbol on your viewer
TROUBLESHOOTING TIPS

Your phone is now configured for the specific Cardboard viewer parameters.

Developing apps for Cardboard

At the time of writing this article, Google provides two SDKs for Cardboard:

- Cardboard SDK for Android (https://developers.google.com/cardboard/android)
- Cardboard SDK for Unity (https://developers.google.com/cardboard/unity)

Let's consider the Unity option first.

Unity (http://unity3d.com/) is a popular fully featured 3D game engine, which supports building your games on a wide gamut of platforms, from Playstation and XBox, to Windows and Mac (and Linux!), to Android and iOS.

Unity consists of many separate tools integrated into a powerful engine under a unified visual editor. There are graphics tools, physics, scripting, networking, audio, animations, UI, and many more. It includes advanced computer graphics rendering, shading, textures, particles, and lighting with all kinds of options for optimizing performance and fine tuning the quality of your graphics for both 2D and 3D. If that's not enough, Unity hosts a huge Assets Store teaming with models, scripts, tools, and other assets created by its large community of developers.

The Cardboard SDK for Unity provides a plugin package that you can import into the Unity Editor, containing prefabs (premade objects), C# scripts, and other assets. The package gives you what you need in order to add a stereo camera to your virtual 3D scene and build your projects to run as Cardboard apps on Android (and iOS).

If you're interested in learning more about using Unity to build VR applications for Cardboard, check out another book by Packt Publishing, *Unity Virtual Reality Projects* by Jonathan Linowes (https://www.packtpub.com/game-development/unity-virtual-reality-projects).

So, why not just use Unity for Cardboard development? Good question. It depends on what you're trying to do. Certainly, if you need all the power and features of Unity for your project, it's the way to go.

But at what cost? With great power comes great responsibility (says Uncle Ben Parker). It is quick to learn but takes a lifetime to master (says the Go Master). Seriously though, Unity is a powerful engine that may be an overkill for many applications. To take full advantage, you may require additional expertise in modeling, animation, level design, graphics, and game mechanics.

Cardboard applications built with Unity are bulky. An empty Unity scene build for Android generates an *.apk* file that has minimum 23 megabytes. In contrast, the simple native Cardboard application, *.apk*, is under one megabyte.

Along with this large app size comes a long loading time, possibly more than several seconds. It impacts the memory usage and battery use. Unless you've paid for a Unity Android license, your app always starts with the *Made With Unity splash* screen. These may not be acceptable constraints for you.

In general, the closer you are to the metal, the better performance you'll eke out of your application. When you write directly for Android, you have direct access to the features of the device, more control over memory and other resources, and more opportunities for customization and optimization. This is why native mobile apps tend to trump over mobile web apps.

Lastly, one of the best reasons to develop with native Android and Java may be the simplest. You're anxious to build something now! If you're already an Android developer, then just use what you already know and love! Take the straightest path from here to there.

If you're familiar with Android development, then Cardboard development will come naturally. Using the Cardboard SDK for Android, you can do programming in Java, perhaps using an IDE (integrated development environment) such as Android Studio (also known as IntelliJ).

As we'll notice throughout this article, your Cardboard Android app is like other Android apps, including a manifest, resources, and Java code. As with any Android app, you will implement a *MainActivity* class, but yours will extend *CardboardActivity* and implement

CardboardView.StereoRenderer. Your app will utilize OpenGL ES 2.0 graphics, shaders, and 3D matrix math. It will be responsible for updating the display on each frame, that is, rerendering your 3D scene based on the direction the user is looking at that particular slice in time. It is particularly important in VR, but also in any 3D graphics context, to render a new frame as quickly as the display allows, usually at 60 FPS. Your app will handle the user input via the Cardboard trigger and/or gaze-based control.

That's what your app needs to do. However, there are still more nitty gritty details that must be handled to make VR work. As noted in the Google Cardboard SDK guide (https://developers.google.com/cardboard/android/), the SDK simplifies many of these common VR development tasks, including the following:

- Lens distortion correction
- Head tracking
- 3D calibration
- Side-by-side rendering
- Stereo geometry configuration
- User input event handling

Functions are provided in the SDK to handle these tasks for you.

Building and deploying your applications for development, debugging, profiling, and eventually publishing on Google Play also follow the same Android workflows you may be familiar with already. That's cool.

Of course, there's more to building an app than simply following an example. We'll take a look at techniques; for example, for using data-driven geometric models, for abstracting shaders and OpenGL ES API calls, and for building user interface elements, such as menus and icons. On top of all this, there are important suggested best practices for making your VR experiences work and avoiding common mistakes.

An overview to VR best practices

More and more is being discovered and written each day about the dos and don'ts when designing and developing for VR. Google provides a couple of resources to help developers build great VR experiences, including the following:

- *Designing for Google Cardboard* is a best practice document that helps you focus on the overall usability as well as avoid common VR pitfalls (http://www.google.com/design/spec-vr/designing-for-google-cardboard/a-ne...).
- *Cardboard Design Lab* is a Cardboard app that directly illustrates the principles of designing for VR, which you can explore in Cardboard itself. At Vision Summit 2016, the Cardboard team announced that they have released the source (Unity) project for developers to examine and extend

(https://play.google.com/store/apps/details?id=com.google.vr.cardboard.ap... and https://github.com/googlesamples/cardboard-unity/tree/master/Samples/Car...).

VR motion sickness is a real symptom and concern for virtual reality caused in parts by a lag in screen updates, or latency, when you're moving your head. Your brain expects the world around you to change exactly in sync with your actual motion. Any perceptible delay can make you feel uncomfortable, to say the least, and possibly nauseous. Latency can be reduced by faster rendering of each frame and maintaining the recommended frames per second. Desktop VR apps are held at the high standard of 90 FPS, enabled by a custom HMD screen. On mobile devices, the screen hardware often limits the refresh rates to 60 FPS, or in the worst case, 30 FPS.

There are additional causes of VR motion sickness and other user discomforts, which can be mitigated by following these design guidelines:

- Always maintain head tracking. If the virtual world seems to freeze or pause, this may cause users to feel ill.
- Displays user interface elements, such as titles and buttons, in 3D virtual spaces. If rendered in 2D, they'll seem to be "stuck to your face" and you will feel uncomfortable.
- When transitioning between scenes, fade to black, cut scenes will be very disorienting. Fading to white might be uncomfortably bright for your users.
- Users should remain in control of their movement within the app. Something about initiating camera motion yourself helps reduce motion sickness.
- Avoid acceleration and deceleration. As humans, we feel the acceleration but not constant velocity. If you are moving the camera inside the app, keep a constant velocity. Rollercoasters are fun, but even in real life, it can make you feel sick.
- Keep your users grounded. Being a virtual floating point in space, it can make you feel sick, whereas a feeling like you're standing on the ground or sitting in a cockpit provides a sense of stability.
- Maintain a reasonable distance from the eye for UI elements, such as buttons and reticle cursors. If too close, the user may have to look cross-eyed and can experience an eye strain. Some items that are too close may not converge at all and cause "double-vision."

Building applications for virtual reality also differ from the conventional Android ones in other ways, such as follows:

- When transitioning from a 2D application into VR, it is recommended that you provide a headset icon for the user, the tap, as shown in the following image:
- To exit VR, the user can hit the back button in the system bar (if present) or the home button. The cardboard sample apps use a "tilt-up" gesture to return to the main menu, which is a fine approach if you want to allow a "back" input without forcing the user to remove the phone from the device.

- Make sure that you build your app to run in fullscreen mode (and not in Android's Lights Out mode).
- Do not perform any API calls that will present the user with a 2D dialog box. The user will be forced to remove the phone from the viewer to respond.
- Provide audio and haptic (vibration) feedback to convey information and indicate that the user input is recognized by the app.

So, let's say that you've got your awesome Cardboard app done and is ready to publish. Now what? There's a line you can put in the *AndroidManifext* file that marks the app as a Cardboard app. Google's Cardboard app includes a Google Play store browser used to find a Cardboard app. Then, just publish it as you would do for any normal Android application.

Summary

In this article, we started by defining Google Cardboard and saw how it fits in the spectrum of consumer virtual reality devices. We then contrasted Cardboard with higher end VRs, such as Oculus Rift, HTC Vive, and PlayStation VR, making the case for low-end VR as a separate medium in its own right. We talked a bit about developing for Cardboard, and considered why and why not to use the Unity 3D game engine versus writing a native Android app in Java with the Cardboard SDK. And lastly, we took a quick survey of many design considerations for developing for VR, including ways to avoid motion sickness and tips for integrating Cardboard with Android apps in general.

Resources for Article:
