Engines and Frameworks for 2D Game Development

Created for GameDevNation.com
by J. A. Whye, Three Ring Ranch

While there are literally dozens of engines and frameworks available for creating 2D video games, there are a few that stand out for different reasons. In this report I'll cover some of the main pros and cons for two particular tools:

- **Unity** http://unity3d.com
- **Corona SDK** http://coronalabs.com

Both of those tools are free to use and both have paid options available if you want more features (most games can be completed and published with the free versions). I've picked those two because I have more than a little experience which each — I've created courses for both and have used each one to publish multiple games for desktop and mobile devices. I'll also briefly look at Defold and Marmalade Quick, two newer tools that you may want to keep an eye on for possible future use.

How the Ratings Work

Each of the engines/frameworks are rated from 1 to 5 (higher numbers are better) in the following categories:

- **Learning Curve** - How fast can an experienced programmer become productive?
- **Rapid Development** - How quickly can typical game mechanics be implemented?
- **Open-Ended** - Are there limitations, or can we build any kind of 2D game?
- **Targets** - Which platforms can you target for the games you make?
- **Documentation** - Is good documentation available, including 3rd-party resources?
- **Community** - Is there a vibrant community available for questions, sharing info, etc?
- **Extras** - Special features that make the tool nice to use.

The ratings are subjective, based on my experience with each tool.

Unity

Unity came to market as a 3D engine, but people have been using it for 2D development for years. As of Q4 2014, Unity Technologies added tools that made 2D a "first class citizen" of the engine and its use for 2D games has exploded.

Unity is a complete engine with a visual-based IDE that's used to create games. While code (either C# or a derivative of Javascript called Unityscript) can be written using any code editor, the IDE in Unity must be used to create the non-code parts of the game.

Because Unity was designed for 3D development, there are many parts of the tool that are not needed for 2D development — and that sometimes make it seem as if you're using a sledgehammer to kill a gnat. You not only need to learn the API for Unity, but also how to use the tools in the IDE (and which ones to ignore when doing solely 2D games), which means the learning curve for Unity is quite steep, even if you already know C# or Javascript.

Once you have a handle on the engine and the IDE it's pretty quick to put something together. Sprites, or game objects, are made of components such as colliders, images, and scripts. The ability to save a created game object as a "prefab" (prefabricated object) and instantiate it as many times as you want makes it easy to put building blocks together that can speed up your game development.

Unity has been used to create just about every kind of game under the sun, from casual puzzle games to platformers to racing games. Games in every genre have been created using Unity, and for every targeted platform. Unity is now able to target every major (and most of the minor) platforms including mobile, desktop, console, smart TVs, VR headsets, etc..

Probably the biggest advantage of using Unity is that it's the leading full-featured game engine with about 45% market share (http://unity3d.com/public-relations). This means the 3rd-party market is booming and there's always someone who can answer questions.

My total rating for Unity over all categories is 30 — a break down of the ratings is provided below.

Corona SDK

A framework rather than a complete engine, Corona SDK has been around for about 5 years and has made a name for itself in the 2D mobile space as a tool that allows rapid game

development. This, despite the fact that Corona is code-only (using the language Lua) — there's no official IDE or visual editor for it.

The lack of a visual tool might be one reason Corona SDK scores so high in the learning curve rating. Someone with Lua programming experience can have something on the screen and moving under player control in under an hour. And Lua is so quick to pick up that an experienced programmer in another language could be up and running in a day or so.

Programmers who come from a visual layout background may roll their eyes at the idea of a code-only framework, but once they've become acquainted with the API they discover it's not just the learning curve that's easy — getting the foundation of a game up and running can be done very quickly wth Corona SDK. Fast development is the main reason I personally keep one foot in the Corona world even while being paid to use Unity.

The main downside I see with Corona is that it's kind of a closed system. The API is massive and it's easy to do everything — everything that Corona allows you to do. If you want to do something the API doesn't provide, you're out of luck. You could buy the Enterprise version and code additions to the framework, but if you stick with the free version you have to know there may be roadblocks ahead.

That said, as long as you know the limitations going in you won't be surprised. And there are tons of games that can be created using Corona SDK. If you need to squeak every bit of speed from the computer in order to create a hard-core racing game, or something similar, I'd say Corona isn't a good fit. But for casual puzzle games, adventures, RPGs, strategy games, etc., it can handle all of those genres (and more) perfectly well.

Corona SDK started out as a mobile device only framework, but now also allows you to target Mac and Windows desktop as well as tvOS (Apple TV).

The documentation for Corona is well-written and the community, while small, is very active and willing to give help. The engineers of Corona Labs often answer questions in the forums.

My total rating for Corona SDK over all categories is 30 — see the table below:

2D Game Development Tools (larger numbers are better)

	Unity	Corona SDK
Learning Curve	1	5
Rapid Development	4	5
Open-Ended	5	3
# Target Platforms	5	3
Documentation	5	4
Community	5	5
Extras (Mostly Subjective)	5	5
Total	30	30

While Unity received more perfect scores of 5, the steep learning curve rating pulls the overall score down enough that we ended with a tie. And I think that's a pretty fair assessment of both tools — no matter which one you choose, you'll end up with something really good.

The Up-and-Coming Tools

The following game tools are ones to keep an eye on. While I haven't used either one very much, I've poked at both enough to be impressed with them for different reasons. Both are also available for free, and target at least the major mobile and desktop platforms.

Defold

Defold is a new 2D engine that was bought by King (of Candy Crush fame) and released for free in Q1 2016 as an open beta. It uses Lua for the scripting language, but also includes a full-featured visual IDE. It's somewhat like Unity, but aimed at 2D development (the engine is actually 3D-capable, but the tools for doing 3D don't yet exist).

Game objects in Defold are reminiscent of game objects in Unity — they are made up of different components and can be saved and reused. One useful feature built into Defold is the ability to easily collaborate with others on your team.

Marmalade Quick

Marmalade Quick (MQuick) is a lot like Corona SDK in that it's a framework and not an engine — it's all code-based. Marmalade is a C++ framework built around Cocos2D-X, which has been used to ship hundreds (probably thousands) of games, and Marmalade Quick is the Lua-based version.

MQuick appears to have been created by someone familiar with Corona SDK as many of the APIs are very similar. That makes switching from Corona to MQuick (or vice versa) pretty easy. The learning curve is slightly higher than with Corona, but once you know your way around it's a good framework for getting something working quickly.

(One caveat — the last time I looked into MQuick there was a problem with audio. [Famous game developer who I don't have permission to quote] was using it for a new game and ended up switching to Corona due to those problems, saying, in essence, "…Quick just isn't ready for prime time." As Marmalade is continually updating the framework it's possible those problems have been fixed.)

Summary

Choosing either Unity or Corona SDK for 2D game development could be considered a win. You can get going faster with Corona, but with Unity you can target more platforms.

- If you're an existing game development studio who can choose only one, I recommend Unity simply because there are no limitations to what you may want to work on in the future; for example, console development. *(On the other hand there are studio who are hard-core Corona shops and wouldn't have it any other way.)*
- If you're a hobbyist or smaller game studio who's planning on mainly targeting mobile devices, I recommend Corona SDK for its rapid development. Plus, it's just "more fun" to work with — I know that's subjective, but I've also heard it from others who have experience with both Corona and Unity.

Of course, since the best tool for the job often changes due to the specs of each particular job, there's nothing wrong with becoming fluent in both. You'll then have the best of both worlds. For games delivered to mobile or desktop and with known parameters, using Corona will allow you to deliver the finished product more quickly. For games that need more versatility or more diverse publishing targets, choose Unity.

Engine / Framework Links (alphabetically)

Corona SDK

http://coronalabs.com

Defold

http://defold.com

Marmalade Quick

https://www.madewithmarmalade.com/marmalade/technicaldetails#tab=quick

Unity

http://unity3d.com

Corona SDK Tutorial Links

Mastering Corona SDK

http://masteringcoronasdk.com

Unity 2D Tutorial Links

Mastering Unity 2D

http://masteringunity2d.com