

Technically Speaking—Developing in Unity

Alex Schwartz

My name is Alex Schwartz, technical artist & multi-platform generalist game developer located in Boston. Over the past year or so, I've been developing games and interactive media for Apple hardware (iPhone, iPod Touch, iPad), some of which include published titles available on Apple's App Store. Working primarily in the Unity engine to create these works has been a wonderful experience for me, so I'd like to share my thoughts and review the toolset, features, challenges, and talk a bit about the community.

Initially, I was drawn to Unity over two years ago due to the polish of their integrated editor. At the time, Unity was a Mac-only development package and provided deployment paths for Mac, web, and PC distribution. Now, only a short time later, many more platforms are available including Apple's iDevices. In addition to a polished editor, the allure of a game development package that claimed to be open and friendly to scripters and artists appealed to me at first. Coming from a scripting and tech art background, I wanted something with a shallow learning curve, but with serious depth and flexibility. In my experience, many of the low to medium priced generic game engines out there suffer from a 'rough around the edges' user experience, which turns off a large crowd, including designers and artists. Unity excels in this regard, providing a complete editor suite with all the basic components ready to go: a built in sound engine, physics, a graphical editor, input management, a material and shader editor, and much more. If you're making a 2D or 3D game, Unity gives you a huge advantage right off the bat giving you most of the necessary infrastructure, which allows you to focus on creating your gameplay. Having this infrastructure makes it especially good for rapid prototyping game ideas or enabling game jam participants to create low scope high quality games in a weekend or even a single day.

In terms of programming support, Unity offers three scripting languages: JavaScript, C#, and a dialect of Python called Boo. All three are equally fast, interoperate, and can call upon any .NET libraries to support database access, XML, file access, and networking support. All of the scripts are AOT compiled to native C++ code when deploying to iPhone, and build a fully customizable Xcode project in a single button. Unity also provides functionality for

bi-directional communication with Objective-C code. Various middleware companies such as OpenFeint have created easy integration packages that hook directly to your Unity scripts.

Unity's included components allowed me to develop my iPhone title SpringFling in a few short months. During development, I was able to make use of the built in PhysX physics engine to propel the main character realistically. I used simple API calls to track touch input and calculate the angle and distance of a finger drag event, applying simple forces to the character, which caused them to fly into the air in a specific direction and intensity. Unity's API is well structured and extensively documented, which gives the engine a definite advantage over other competitors that provide engines for iPhone deployment. One design choice by the Unity developers that I find

**"HAVING THIS INFRASTRUCTURE
MAKES IT ESPECIALLY
GOOD
FOR RAPID PROTOTYPING
GAME IDEAS"**

particularly useful is the similarity of the iPhone touch class when compared to the mouse input class. A seemingly complicated task such as porting to a web player, desktop, or Facebook is literally a few lines of platform-specific code away. Unity has been proven successful in the iPhone market, some examples of which include Skee-Ball, Zombieville USA, Ravensword and Star Wars Trench Run (developed by local Boston developers Infrared5) just to name a few. Therefore, being able to expose your game to numerous markets can offer a huge marketing advantage, one that I foresee developers utilizing more extensively as the platform matures. With Android support coming with the release of Unity 3, multiplatform deployment offers more options than ever before.

One of the greatest assets of choosing Unity for development is surprisingly unrelated to the tools themselves. The community around the Unity engine is amazingly strong, with several thousand active posters on the main forums (forum.unity3d.com) and hundreds contributing to a large community wiki (unifycommunity.com) filled with code samples, scripts, shaders, and editor helpers to



B.U.G.
BOSTON UNITY GROUP

streamline your workflow. I've been involved in several engine-specific communities over the years (C4, Torque, Trinigy, and a few others) and I've found Unity's to be the most welcoming, comprehensive, and friendly. Even the CEO himself, David Helgason, is known to answer common questions on the support forums.

Speaking of Unity communities, Elliott Mitchell and I have started up a user group known as the Boston Unity Group (B.U.G.). Our aim is to bring together local like-minded developers into the same room for a bi-monthly meeting with the purpose of sharing experiences, educating one another, and making both casual and business connections. Our first meeting held on June 12th is an all day workshop and info session run by Tom Higgins, community manager at Unity Technologies. With 125 attendees slated to arrive, it's a strong sign of Unity's penetration among local game development circles.

Unity's affordable licensing structure and lack of per-game fees make it a great place to dive into with minimal up-front investment. Developers keen to the recent uproar over changes to the Apple developer license agreement may be asking about the future viability of Unity on iDevices. To summarize, Apple's new verbiage implies that third party toolsets (Torque, Unity, Flash CS5, MonoTouch, and many others) could be banned from the App Store due to the cross-compilation. It's my opinion that Unity will remain a dominant force on the App Store, continuing to create some of the highest quality games available. Apple would be crazy to cut out a large percentage of their developers. Technologies change quickly and allegiance to one toolset usually has its disadvantages, but the strong community and no-BS attitude of the Unity Technologies crew inspires confidence to stick with their platform for years to come. Their mantra of 'Code once, deploy everywhere' seems to be really hitting home with developers looking to the future of mobile and web gaming. With Android support only a short time away, it's a good time to get to know Unity. ■

More information about Alex's iPhone game SpringFling can be found at <http://springflinggame.com>. Alex Schwartz can be contacted via his blog, <http://gtproductions.net/blog> or at twitter.com/gtjuggler.