# Postmortem: Pangalore's Knightly Adventure

PRINT

By Doyon Kim


In this postmortem, we'll take a look at the ups and downs of developing Pangalore's first major title, *Knightly Adventure*. Developing a first big project with a new team is a time of excitement, and we had a fun ride along the way. Sometimes it was the thrill of seeing a plan come together, and at other times it was the panic of seeing a mistake that we'd overlooked -- but the good times outweighed the bad.

At Pangalore, we specialize in multiplatform mobile games using both HTML5 and Unity. Our philosophy is that you should be able to take your game with you wherever you go and play on whatever device you have in hand. Previously, we had put this into practice with simpler HTML5 games such as *Pop the Candy* so we could work on social mechanics and the intricacies of getting games to run well on multiple devices and platforms.

We had also worked on getting our development and publishing pipeline working well across two continents -- our dev team is based in Korea with our publishing team located in San Jose, CA. I split time between both locations, working on our U.S. publishing operations while making frequent trips to interface with the development team and communicate the needs of our international audience. (I have a lot of frequent flier miles if anyone needs points!)

As we sat down to kick off *Knightly Adventure*, a social/action RPG hybrid designed to offer "build and battle" gameplay, we were pretty sure we had something that was unique in the marketplace. The stakes were higher but this would be a big step forward for our company if we could pull it off.

Our value proposition was that players could enjoy the game on multiple devices throughout the day, with all progress saved, and the same 3D graphics available on every device. Unlike some titles where you pay for your game separately on each new platform, our free-to-play adventure would be the same game in every instance, and a purchase made on any platform would carry over to all the others.

Our target platforms included iOS and Android as native apps as well as PC/Mac via Facebook. *Knightly Adventure* is set in a medieval fantasy world where players can build kingdoms, cultivate crops, craft items, and fulfill RPG battle quests with friends. The design was set up so you could play it on your phone during your train ride to work or school, on Facebook on a work PC at lunch, and on a tablet in the evening to relax.

While the game can be played in short bursts, it can also be played in longer sessions that bounce back and forth between simulation and action sequences. For instance, you can plant some crops or start an item build in your town and then venture outside with a party to battle colorful monsters, returning to town to harvest your goodies.

With a multiplatform game with a new team working on two continents, we didn't do everything perfectly along the way. But while we had some bumps along the road with our first flagship title, we put together a very successful launch that saw *KA* rise to the #1 RPG on iOS in 23 countries in terms of downloads. Despite this considerable success, we could have done even better. Our goal in this postmortem is to share both what worked for us and some

mistakes that we'll certainly try to avoid in future titles as we improve our development process.

## What Went Right

**1. It's Two, Two Things in One**

The blend of two distinct genres in *KA* created a fresh style of game. Having social and RPG elements extended both genres without excluding fans of either style of game. On their own, *KA*'s social and RPG features may not have stood out, but as a blend they allowed us to get more attention from press and players alike. (For instance, 148apps.com said, "*Knightly Adventure* is a game that is really the sum of its parts that all come together to make something better than it would be individually -- and its excellent cloud-based saves support helps too.")

Our mantra for the game was "continual engagement" -- a player need never wait for a build to finish. They can set builds, go on an RPG battle quest, and return to harvest crops and materials. Players can select short builds for the time they're online (during which they bounce back and forth with quests) and then longer (and more valuable) builds to fill in the time while they're away from the game.

The RPG quests are designed to be easy to play to lower the barrier for social players while still incorporating strategy. Most quests are designed for a party of three characters and players can include the avatars of their friends to make up the right mix for a given quest. You choose your party with the characters of up to two friends or NPCs (it's asynchronous so you're choosing their avatars and they don't have to be online). Picking the best mix of melee/ranged/magic characters is important for various missions and you also want companions of the right level -- higher-level characters cost more in game currency so players usually want to balance their team for the best cost/performance.


***KA*'s "build and battle" hybrid gameplay**

---

**2. Ride the Feedback Cycle**

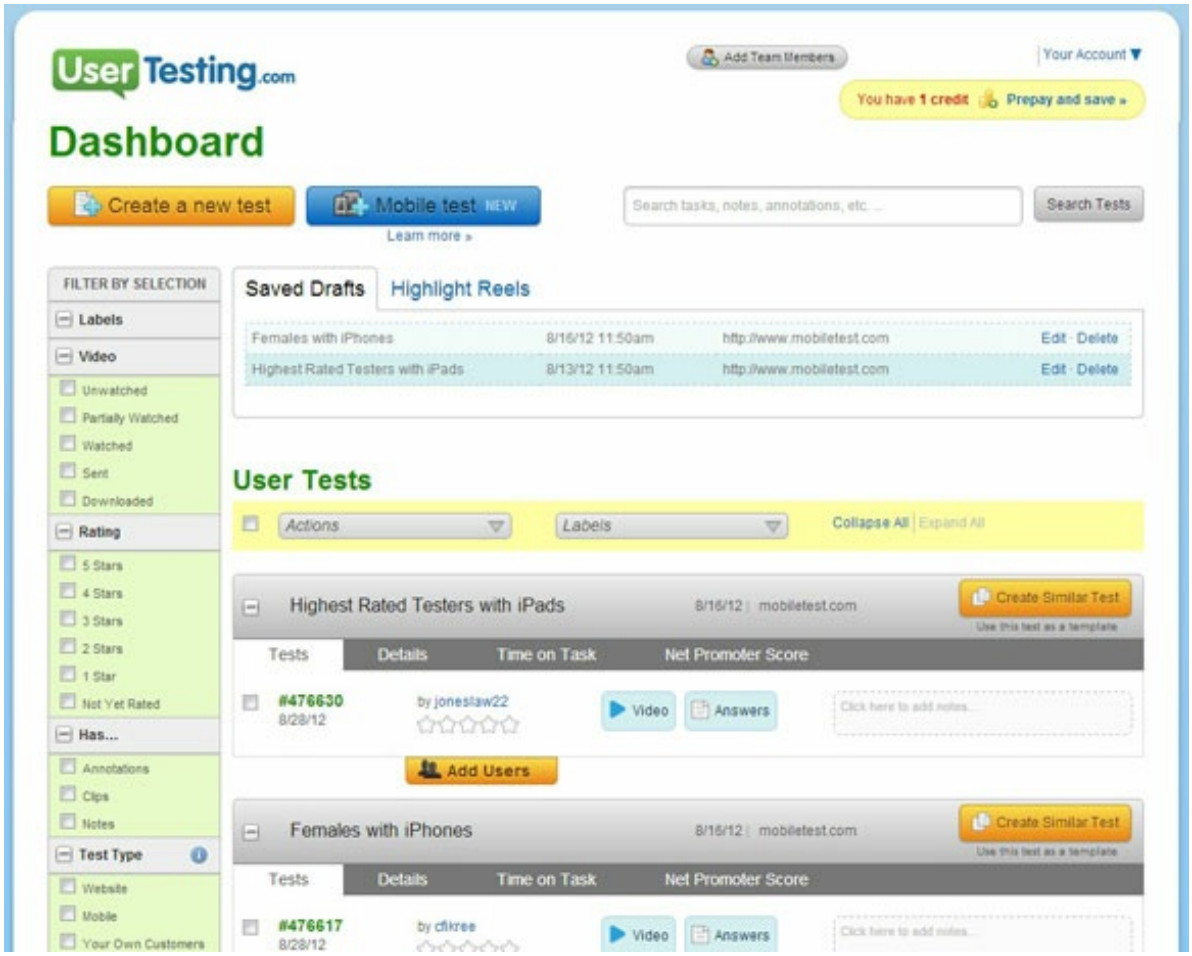
We build multiple feedback cycles into our dev process to allow the team to improve features before launch. Our philosophy is that we don't know everything but we can learn from everyone -- testers, beta players, team members, and press alike. Like most online games we went through multiple beta cycles, a Canadian test launch on iTunes, and we also took feedback from press during guided game tours. Most developers try to do the same, but we emphasized to the team throughout the process that we wanted to take advantage of any useful feedback, no matter what the source. If the janitor had something good to say, we'd listen.

The first and most intense feedback cycle came from the core team, all of whom are people who are committed to the game. This feedback was the most in-depth and continued all through the process, leading us to change or fix many things from battle mechanics to character animations. Our in-house testers in Korea were also helpful at stopping issues. However, because the team also knows the game best, they didn't necessarily see issues that new players

might experience. Sometimes you get used to 'how things work' and ignore glaring issues.

For those issues, our two QA engineers in Korea and our beta testers and our iOS players during the Canadian "test launch" helped spot more problems. We had to focus on issues in the core gameplay as well as problems that could pop up on one device or another, but we kept a good attitude throughout the team toward anything we felt could improve the game. When we had builds we could share more widely, including with members of the gaming press, we got even more valuable feedback -- the only drawback there was that some was too late in the cycle to implement by launch.

Post-launch, we of course listened closely to user feedback and are using that to shape some of what we aim for in future updates. And of course we watch our metrics closely, similar to other social developers. Crash reports are particularly important for allowing us to hone in on any critical user issues but of course we also look at retention, length of play session, etc. to see how players are enjoying the game. However, we found most crash tracking tools do not support the Unity application very well.




***Knightly Adventure* testing dashboard (click for larger version)**

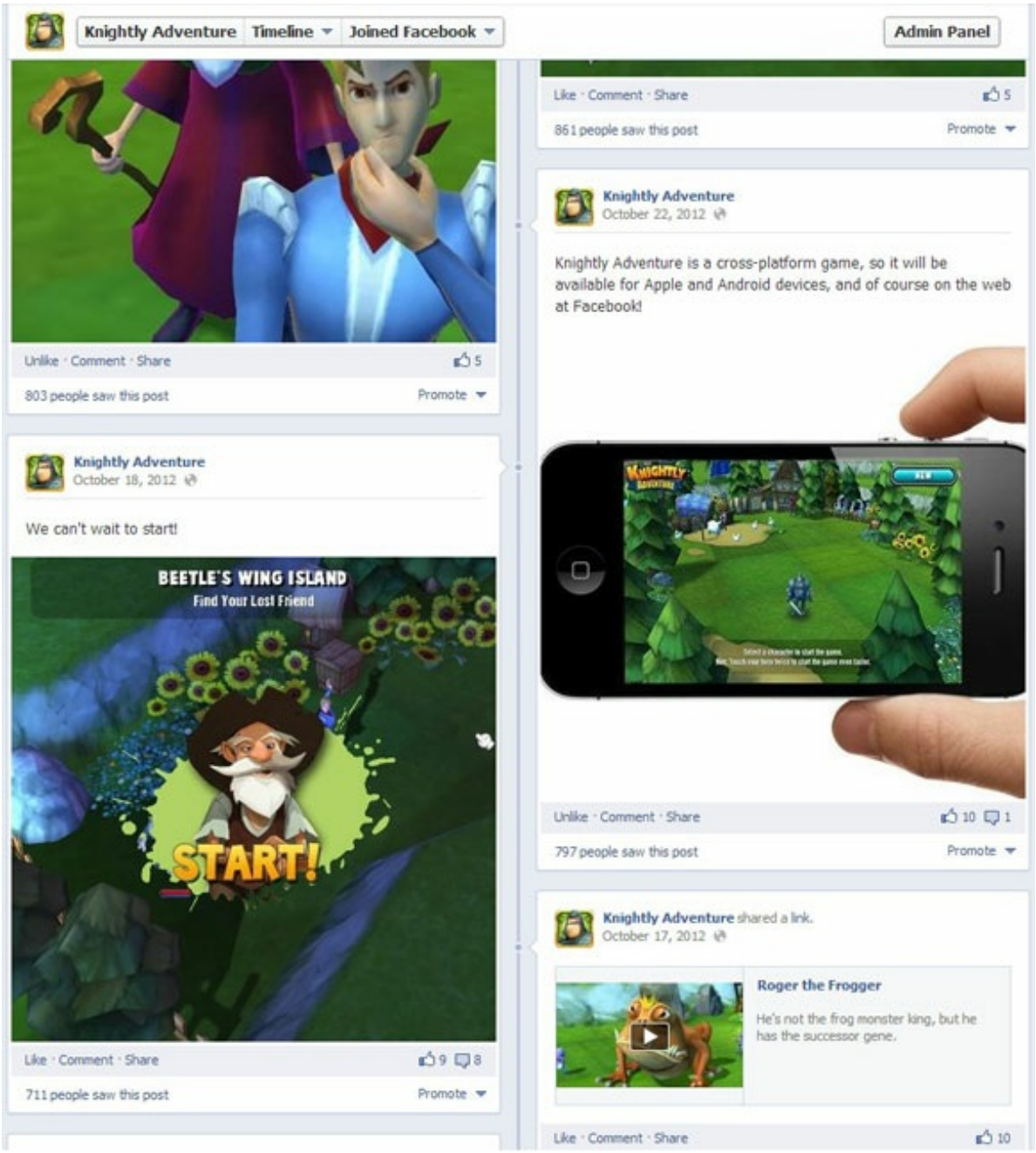
**3. Community and PR Helped Spread the Word**

We didn't launch *KA* when we initially planned -- we were months later than anticipated, in fact. But the upside of this (making lemonade from lemons) was that we had more time to utilize community and PR throughout the cycle to get the word out about the game early enough to create awareness at launch. We were able to establish demand without spending a ton on marketing. This is very valuable for smaller companies like us that don't have big budgets.

We launched our social media campaign on Facebook months before launch, attracting a small but dedicated audience who helped us spread our message. Now sometimes all fans were saying was "when are you going to

launch your game?" but this still gave us a chance to engage them in conversation. Even when we didn't have a new item or asset to reveal, we'd talk about other things that could relate to the game even if it was just a picture of a frog riding a squirrel (frogs feature prominently in the game).

Our PR campaign started with press meetings at GDC in 2012, over six months before our eventual launch. Because the launch was delayed, we did end up with some "dead time" in our PR cycle, but we maximized our presence closer to launch when it mattered most. Because assets equate to coverage (and no assets means no coverage!), we dedicated a team member toward creating videos and screens to maximize PR coverage. This person was on the publishing side rather than the dev team since we didn't want to pull anyone from dev tasks, but it was still a cost against the bottom line -- and one we were happy to pay. We were able to get some nice previews ahead of launch to help set the stage for our appearance in the App Store.




**Reaching a core audience before launch**

---

**4. Partnerships Pay Off**

As a self-publisher it's key to invest in relationships -- there are ten million other games out there so you can't expect everyone to immediately realize why YOUR game is so special compared to everyone else's. You have to make your case well and respect your partners to get traction when you're an indie developer. From Apple to Unity to press, we worked to make our case well ahead of our launch. Not every relationship paid off in a big way, but we look at relationships as a long-term investment that can pay off down the road.

Realizing the importance of the Apple ecosystem, we enlisted help in making our initial outreach to decision-makers in Cupertino. If you're an unknown quantity like we were, reach out for help from anyone you know who may have connections -- I'm not sure we could have gotten a meeting on our own.

Our presentation and budding relationship helped us later when the game's reception helped us to get both "New and Noteworthy" and "What's Hot" exposure in the App Store, both of which were critical to our success. Now, relationships alone won't put you over the top; you still need a sound concept and good execution. However, if you are a newcomer it's hard to get noticed in the crowded mobile marketplace -- and it's getting harder all the time in our experience. Identifying those relationships that will be most important and having a sound plan of action -- from getting a mutual colleague for an introduction to what you present and how you follow up -- should be part of your launch plan.

**5. Multiplatform Synced Play**

As mentioned earlier, synced play was a big part of our initial strategy. Ultimately, we were able to offer a play style that worked with how players are using multiple devices throughout the day. From day one, we worked from the perspective of what the player wants rather than what might look enticing on a budget spreadsheet. Many publishers make players pay separately on each new device they launch on rather than letting them carry over their progress.

Instead, we followed the player to whatever device they wanted to use, with no penalties. Rather than being tied to a device, our game is tied to a PLAYER and the player can choose to play wherever or however they want. Our paradigm is playing on smartphone while commuting, on PC at work during lunch, tablet while relaxing at home in the evening -- different devices for different situations but the same game the whole time.

*KA* is server-based, so game progress is constantly updated on the server no matter which device the player is using. We use the player's Facebook account as the login for syncing game data across devices, and as long as they use the same Facebook ID their game will always be synced on every device. Even if you lose your connection while playing, your progress will be saved up to that point. Of course, you need a network connection to take advantage of this feature but being available on multiple devices helps the player get more out of the game, and also helps him or her find new friends to play with. If your friends can play across iOS, Android, PC, and Mac, it's easier to connect with them than if they were forced to own a particular device to participate.

Now, some players don't like using Facebook (not everyone wants to have a Facebook account), so we do allow a solo mode for players who don't want to log in so they can play on single device without a Facebook account. However, using Facebook IDs provided a vehicle for offering PC/Mac play as another synced platform so players can, for instance, play on their PC via Facebook at work during a break. Our interface is smoother with touch input, but the mouse still works fine.

Using Unity (as we did for *KA*) or HTML5 helps us build a game for multiple devices quickly, but it still requires fine-tuning for each environment. Whenever we update a game, we have to update 3 different versions (for PC platforms, iOS, and Android) at the same time, and it can be a maintenance burden. So our multiplatform synced play does require extra work but it expands our audience (more potential players across all supported devices) while letting individual players get more out of the game (they can play simultaneously on all the devices they own).

---

## What Went Wrong

Okay, we've talked about the good stuff so far. Now let's delve into the other side of spectrum. We've identified our five biggest missteps in the following sections.

**1. Multiplatform Launch Out of Sync**

We discussed the success we had with our synced multiplatform gameplay above. But one area we stumbled on was on the launch timing between different devices. This occurred because we didn't fully understand the differences between the App Store and Google's Play store and their associated hardware.

The App Store has a longer approval cycle, leading to a longer cycle not only in development but for updates. With the ability to issue quick fixes, the Play store is more suited to multiple iterations. Ideally, we might have launched first on Android where we could release hot fixes to resolve issues before launching on the larger iOS market. Because we worked hardest on the biggest platform launch (iOS), our Android version was not ready at launch and when launched, did not have full device coverage. This meant the marketing dollars we spent at launch were leveraged less effectively since we couldn't send potential players to an Android download. It did leave us with a stronger story for iOS players but we lost our multiplatform message in the process.

To compound matters, we underestimated the time required to tune the game for each supported Android device, further delaying our Android launch. And when we did launch, it was only for a few devices, which was not pleasing (understandably so) to Android players. We're still slowly filling out our roster of Android devices.

On the positive side, we did get the game out for the holidays (we launched in November 2012) in time for a positive reception before the market was inundated with holiday shopping and we had our greatest success on the larger/more unified iOS platform. Going forward, we have realigned our expectations for the two stores and have a plan for utilizing the strengths of both. In some cases, quick is good and in others, slow and careful is the right play.

**2. Testing Was Too Localized**

Unfortunately, our testing did not reveal some key issues in our largest market -- North America. By conducting QA where our devs were (Korea) rather than where most customers were (U.S.), we failed to spot some tech issues that were an issue at launch. That was a big oversight, and it bit us in the end like a dog going after a mailman!

South Korea has much faster and better-integrated internet infrastructure than does the US, and this made us underestimate some issues we would have in the US with downloading and syncing performance at launch. Downloads that seem fine at 20 Mbits per second are much more vexing at 1.5 Mbits/second. During spot-testing in the U.S., we thought the issue was that the development servers we were playing from were based in Korea, but the issue turned out to run deeper than that.

Going forward, we are doing QA in North America using a service company. This gives us another channel for external feedback while also insuring that testing is closer to the reality experienced by our players. Lesson learned!

---

**3. Cultural Differences Matter**

Cultural differences between the dev team in Korea and marketing team (and audience!) in U.S. took some work, especially for a game that uses humor so much. Humor can be very regional and this is an ongoing focus for improvement. Do the jokes carry over? That's a hard one to objectively gauge at times.

We also have to look at player expectations. In Korea, a certain amount of grinding is viewed as absolutely proper, bu here in the US players view this gameplay style as boring. So that's another thing we have to watch out for (and we did get dinged for having repetitive gameplay, which we are working on improving through increased variety in our battle quests).

One example of cultural differences was the cute look for the turtle monsters in the game. North American players

expressed they were not comfortable killing cute turtles, so we made them more mechanized and "bad" looking to alleviate this concern. Interestingly, players had no problem with the cute frog enemies -- perhaps people just care more about turtles!


**Turtle evolution after feedback from U.S. players**

We also now have a U.S. team that gives us feedback on our story and text assets and assists with text localization -- you have to go deeper than just translation to make sure the humor carries across. Our Canadian external QA team will also give us a second opportunity to note cultural issues in localized text.

This sounds obvious, but it's something we should have focused on earlier in the cycle for best results.

**4. Money on the Table**

We focused on gameplay at the expense (no pun intended) of monetization. Our goal was to make the game fun first and worry about optimizing monetization later. This isn't an issue that players are complaining about, but our approach did leave dollars on the table. For instance, we originally made it harder to level up, requiring players to have better weapons and armor that would take players more work in game unless they wanted to take a shortcut and buy them outright. But we decided this approach made non-paying players work too hard, making the game less fun, so we adjusted to make it easier to level up.

As a first-time developer in this marketplace, we had to choose our priorities, and that was a factor in deciding to focus on making the game fun first and foremost. The best monetization techniques will not help if you haven't attracted players and given them a good experience in the first place. So wanted to build an audience, and we've had success in that area.

Going forward, we are seeking to find effective ways to monetize play without unbalancing the game. This sounds great, but the hard part is doing it right! For instance, right now you must choose your health and mana potions before a battle, using a limited amount of slots (more slots can be purchased). Allowing players to purchase more potions if they run out in the middle of a quest could provide revenue. Right now, players are essentially being punished if they don't plan ahead. Providing a quick mid-quest purchase opportunity that could help them finish their quest rather than failing and having to load more potions in inventory and start over is a valid option. We want to find these optional opportunities where he can help paying players without impinging on the game flow for everyone else. We are sensitive to the 'pay to win' perception of F2P games and we always want to err on the side of caution in that regard.

Another area we plan to work on is adapting our premium weapons and armor in line with expectations from the community. Humor has always been a big part of KA and this is reflected in our premium items. You can buy an umbrella or a giant fork to use as a weapon, for example (and it does look funny when a big knight is whomping monsters with an umbrella). We have to be creative, though, because we cannot offer a huge variety of items due to the memory constraints of mobile devices.

---

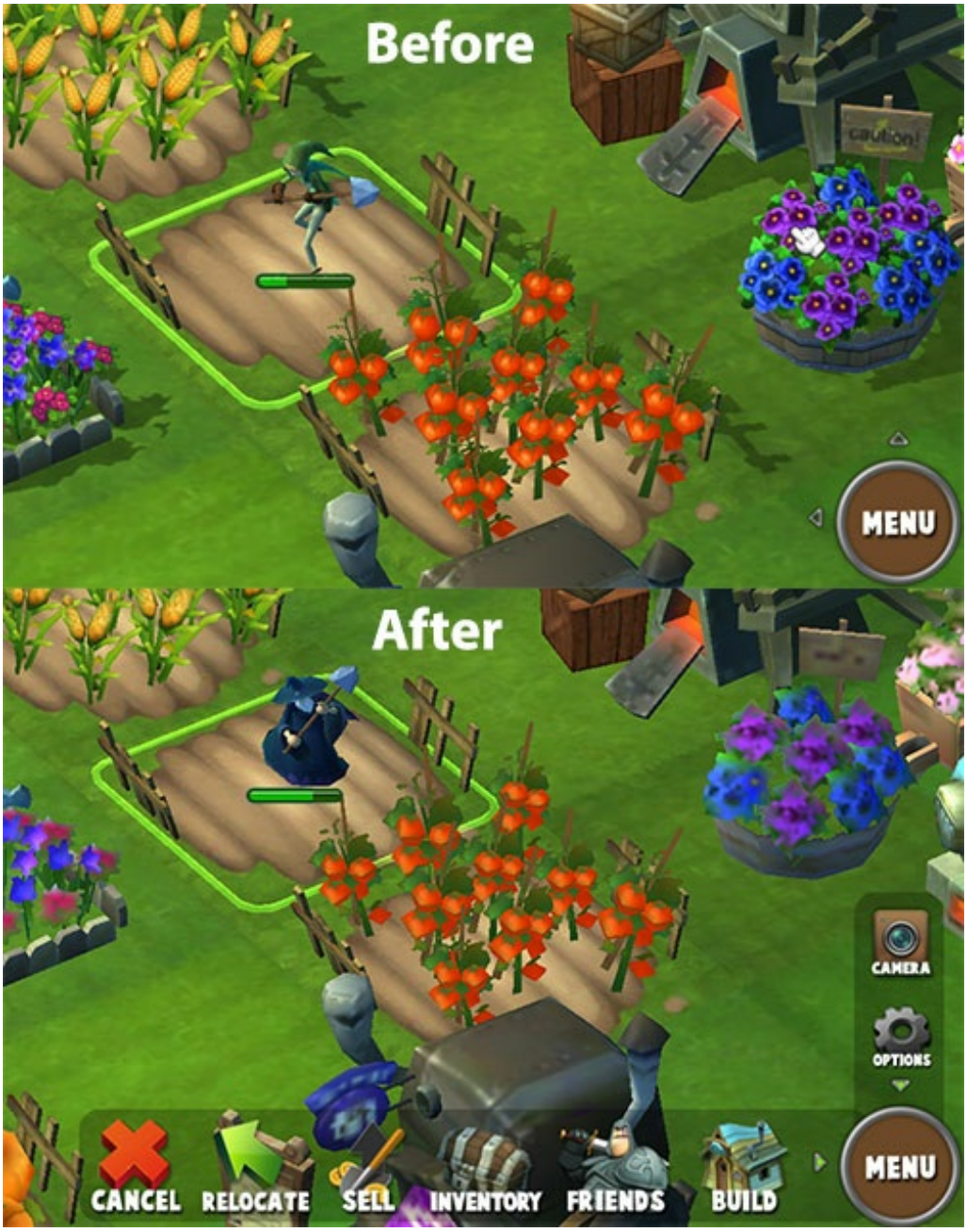
**5. Skewed Focus on the High End**

Our game design focused on high-end graphics, leading to performance issues at launch, especially on older mobile devices. The takeaway for us was to be brutally honest about which devices will work with your game and target your game experience toward those devices. As both the iOS and Android markets continue to be fragmented, this is a critical issue. If you have a 3D action-oriented game like ours you can run into memory and performance bottlenecks very easily.

We knew as we approached launch that we had problems with devices like iPod touch 4th generation and iPhone 3 -- they had worked under iOS 5 but with the iOS 6 upgrade and our own additions we were running into memory crashes. Unfortunately, in our case we could not change the supported iOS platforms text in our iTunes description. Devices such as iPod touch 4 that were now problematic were baked into our app information as being supported.

As a workaround, we updated our main iTunes description and added device checks to warn players that they would experience problems. If a player with an unsupported device tries to load the game they get a pop up warning advising them of this and the game exits so they can uninstall it. However, this was not ideal and some players were understandably upset when they game they had downloaded would not run!

With our most recent update, the issues were largely fixed by compressing our graphics much more heavily. Our environments are not as pretty but ratings are trending toward 4 stars on iTunes. Our next update will restore some graphical sheen with improved resolution and memory optimization so things will get even better (we are stealing some animation frames that are not necessary to the play experience, among other tweaks). However, would have been better to fix this BEFORE launch.




**Note higher compression in the 'after' screenshot**

## Conclusion

With a multi-platform, cross-genre game developed in an international environment, we gave ourselves a high degree of difficulty for our first flagship game. Part of the fun of developing games is constantly challenging yourself, though, and we have no regrets for trying to do something different.

Even though our team was battle-tested in PC online development, the combination of developing in Unity for mobile platforms and trying to support a large range of devices brought fresh challenges. While veteran mobile developers may say, "I told you so!" when they see some of our missteps, there's no substitute for first-hand experience and our team got plenty that will be useful in the future. And hopefully some of our experiences will help other mobile teams do an even better job on their first titles!

Fortunately for us, we were able to achieve a good degree of success with our first title and we hope to achieve even better player experiences with our upcoming titles. Our next game, *ZooVale*, takes some of play mechanics from

*Knightly Adventure* (although in a completely different world populated with cartoon-style talking animals) while executing them in HTML5 for a less graphically intense experience that will run in any browser, including mobile browsers, as well as on iOS as an app. We're excited to see how far we can push HTML5 gaming and thus far we happy with how much performance we can generate through our new HTML5-based engine. And we are still in the early phases of our *Knightly Adventure* -- we are going to keep updating it to make it better by applying everything we have learned so far.

Return to the full version of this article