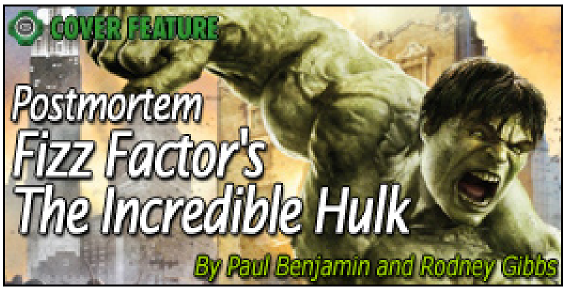
GAMASUTRA
The Art & Business of Making Games

PRINT

Postmortem: Fizz Factor's The Incredible Hulk

By Rodney Gibbs,Paul Benjamin



[In an in-depth, honest Nintendo DS game postmortem, the creators of The Incredible Hulk game discuss making a 'fully destructible environment' title for handhelds, from GameMaker prototyping to 'Rage' button removal.]

Fizz Factor tried something new with the Sega-published *The Incredible Hulk* for the Nintendo DS. Taking our cue from the latest Hulk film, upon which our game was based, we aimed to distill the big green guy down to his essence -- smashing everything.

No stealth, no wimpy Bruce Banner. 24/7 breaking stuff, period. Fully destructible environments -- on a console title this may be an old hat, but on the DS, it's an impressive novelty.

As far as we knew at the time, fully destructible environments had not been tackled before on the DS. Due to cart space and hardware limitations, it's always a big challenge for the platform. However, with some clever programming solutions and rearrangement of budgeted art resources, we conquered the problem, which thrilled us.

To our publisher and the public, however, the innovation did not resonate as we expected. While fully destructible environments are technically impressive and a first for the platform, in hindsight we probably would have been better served to take a more conventional approach that required less engineering and focused more on rich, not-all-that-destructible environments.

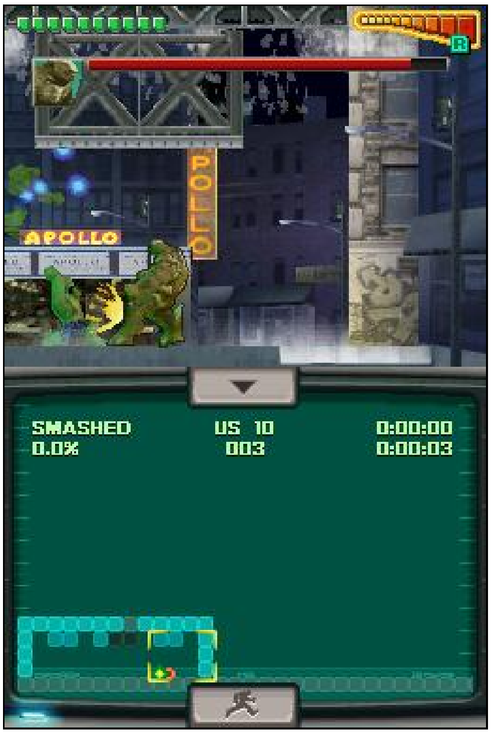
Such an approach would have triggered other design and production decisions that would have affected the game's outcome.

While the game turned out well and is a minor milestone for the platform, developing the game with fully destructible environments created hardships that could have been avoided.

What Went Right

1. Prototyped out all levels with GameMaker before building them on platform.

Our lead designer was an avid user of GameMaker, a PC-based game engine embraced by developers and hobbyists alike. As the engineers were adapting our engine to support fully destructible environments, our designers prototyped out each *Hulk* level in GameMaker.



Using simple sprites we created or nabbed from other games, we laid out each level, enemy AI and the player package, resulting in a great testing ground for gameplay.

To emulate the DS controller experience, we used a PlayStation 2 controller for its D-pad and four input buttons. (*Hulk* had limited touchpad gameplay. For other titles employing more touchpad, we've used a Wacom tablet and the PS2 D-pad to emulate the DS input with GameMaker prototypes.)

Using these methods, prototyping levels was quick -- a level came online in GameMaker in two to three days -- and had tangible payoffs in pre-production and early production.

For one, the prototype enabled the designers to suss out their design with tools they controlled. From level layout to scripting, the designers created each level, its enemies, and Hulk's behavior with this rapid prototype tool.

Having one designer sorting out problems, puzzles, and behavior before engaging the other disciplines saved time up front and got the team excited since they could see what the full game would feel like within a couple of weeks.

Once the designers were happy with a level and our DS engine was ready, the designers largely recreated their GameMaker levels on the platform. Had we a way to export those levels to DS rather than rebuild them, we would have saved even more time.

Another benefit of the prototype was it helped us communicate our game vision to the client. Because we were revving on the technology behind the fully destructible environments, we did not have a strong playable to show our publisher early on. Instead, we had them play the prototype.

This worked well for the first several cycles, since, as with our team, it quickly conveyed the gameplay and excited our client.

2. Developed a fully destructible world on the DS.

Most of the world in *Hulk DS*'s levels can be destroyed. A tile-based system, the levels contain three states: healthy, damaged, and destroyed. Through punching, smashing and throwing objects, the player destroys the world to open up alternative paths, reveal hidden objects, and just bash the crud out of stuff. Engineering such a system on the DS challenged our programmers and artists alike.

Engineering a fully destructible world, while simultaneously developing the engine for wireless download via the Wii and building the rest of the game, was a significant programming task. However, it was a feat we accomplished successfully.

The four-person engineering team rearranged and reused existing technology wherever possible. Programmers leveraged Maya for level building and Lua for scripting. This allowed the design team to build levels and script enemies and events themselves, thus freeing up the engineering team to continue work on other features.

While the top screen displayed the playfield, the DS's bottom screen featured a scout map. An interactive map of the entire level, the scout map tracked tile destruction in real time.

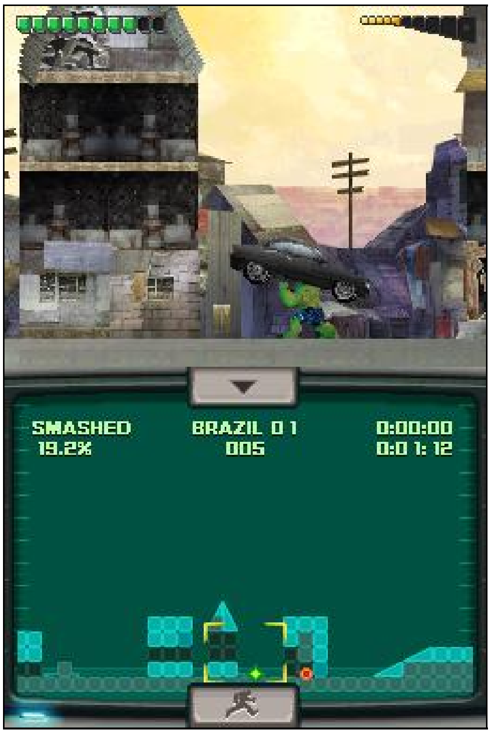
As the player smashed up or down a building in the top screen, for example, the bottom screen tracked that destruction. Dragging a finger across the scout map automatically paused the game while the player peeked at upcoming enemy configurations or plotted routes.

Our environmental artists created three states for each tile to reflect its descent from healthy, to damaged, to destroyed. While some tiles were reused with levels (Times Square, for example, has urban building tiles repeated throughout), creating three states was an immense effort tantamount to creating three times the number of levels in the game. Despite the large workload, our three environmental artists delivered strong work.

While the destructible environments were tile-based, our artists employed clever tricks to obfuscate that fact.

Healthy tiles formed into skyscrapers, for example, are fairly tiled by design. As the player damages a tile, broken elements -- such as jagged bits of rebar, irregular jags of broken glass -- extend beyond the damaged tile's boundaries and into adjacent tiles, be they healthy or damaged, thus avoiding a clearly defined, checkerboard appearance.

In addition, because the artists had created two damaged states for each tile that we randomly summoned, even fully destroyed buildings did not look repeated or tiled.



3. Found positive compromises with the publisher that added to the game.

Throughout the course of the project there was constant discussion with the publisher regarding various game features.

In most cases, we were able to agree on changes that addressed the spirit of the publisher's concerns, while keeping development in scope and adding to the game's fun factor. For example, publisher feedback resulted in knockback, turning enemies into weapons, as well as a wider variety of weapon objects for Hulk to hurl at his foes.

The best example of a positive compromise with the publisher came with changes to the control configuration and the destructible environment. The control configuration was the source of much discussion over the life of the project.

In short, we came to a good compromise to the control configuration that sustained the lead designer's vision while addressing the publisher's desire to give a broader spectrum of players control over Hulk's amped-up Rage attacks and movement.

A side effect of these changes was that Hulk's attacks destroyed objects with a single button press instead of two. While this added to the sense of Hulk's power and simplified the control configuration, it would make one of our three damage states for each tile irrelevant.

Suddenly tiles could go directly from healthy to destroyed, with no need for a damaged state. As developers, we're used to sometimes having to discard work, but this would result in the aforementioned "checkerboard" look to the environment, a concern for our client.

Our solution was to randomize the damage system to choose either the damaged or destroyed state for any destroyed tile. Rather than abandon the tiles' damaged state versions, we leveraged those tiles to create a more varied, less repetitive environment. By doing so, we addressed our publisher's concerns, validated the effort of our environmental artists, and gave the players a better-looking game.

---

4. Communicated the power of the Hulk on the DS.

Over the course of development, the phrase "Feel the power of the Hulk" was our mantra. At every stage of production we kept in mind that Hulk needed to feel like an engine of destruction. The design of the game called for Hulk to be fighting enemies at a distance as well as up close, which necessitated a certain amount of space around him. On a device as small as the DS, that left us working to make a half-inch tall green guy feel like a powerhouse.

Just about every element of the game lends to that sense of Hulk's power. Sound effects were a huge bonus on this front, from footsteps that boomed with every step to powerful impact sounds from smashed objects. The fact that Hulk could destroy everything he could touch went a long way -- as did his abilities to knock enemies into each other, smash vehicles then hurl them at foes, and to turn giant boulders, tree trunks, and cars into weapons.

We also implemented two special features to add to Hulk's sense of power. One was a feature called "Gamma Boost"; while utilizing this feature Hulk essentially emitted a field of destruction that killed or destroyed everything near him, encouraging the player to run through the level with reckless abandon. The more Hulk destroyed, the longer his destructive state would last; however, if he took one hit, the Gamma Boost would deactivate.

The other feature was a series of "Rage Vaults." By grabbing onto a flagpole or a floating gamma detection satellite, Hulk could hurl himself through the world on a set path, destroying everything in his way. This feature not only added to Hulk's power, but also gave the designers some unique opportunities to create platforming puzzles by which the player could navigate the levels.

What Went Wrong

1. The gameplay came together late due to new technology implementation.

With our studio having developed numerous DS games, we had engines to support a variety of game genres for the platform in both 2D and 3D. Since none had tackled fully destructible environments, however, we had to undertake extensive engineering work up front to prepare. The benefits of achieving this goal did not end up outweighing the costs.

While the engine changes were underway, the art and design team could not develop on target. This affected our relationship with the client since it was relatively late into production before we had a solid build to show.

On DS projects in which we built upon existing technology, we have had working -- sometimes spectacularly so -- prototypes on platform during pre-production. On those projects, we instilled confidence with the client early on, which made everyone's job easier.

On *Hulk*, because we developed new technology for the game, the on-target build came later in the process, causing our client to grow impatient. The delay hindered the process for the remainder of the project.

On subsequent projects we have refrained from developing new technology, however impressive we think it might be. We've chosen to rely on proven tech during the dev process, with improvements and innovations to the tech coming from outside the project critical path.

2. Creating the technology to support destructible environments eroded time available for level creation and polish.

Again, the upfront tech investment limited our polish time. Rather than work at a measured pace from the start, we were pushing hard to get all content in the game and be feature complete by alpha and beyond.

While our client was enthusiastic about the fully destructible aspect of our design, the detracted polish time that resulted from its implementation was a warranted source of frustration on both sides.

---

3. Asset creation consumed far more time than we anticipated.

The tile-based system we devised called for artists to create three tiles for each square of gameplay. While this wasn't three-times the overall amount of art we would have created for other side-scrollers, it was a considerable art hit and technical challenge. We were fortunate to have a great environmental artist with excellent tech problem-solving skills.

Working with the programmers, he devised a system for implementing the tiled elements within the memory constraints in a way that kept them from looking tiled. He also created a first pass on all of the tiles. However, the chore was immense, stressful, and is not anticipated to be repeated.

Because the environmental art demands were far more than we anticipated, we had to add two additional artists to the backgrounds from alpha till the end. Taking the placeholder tiles created by the initial artist, they shaped them up and extended the blurring effect between the tiles to occlude any checkerboard effect.

The result, as noted above, was good, since the worlds looked good. The commitment from these artists was Herculean and not something we will or could repeat. Having committed ourselves to the fully destructible world, however, we had no choice but to add more staff and impact our budget.

4. Indecision over control configuration delayed production.

Our lead designer devised an innovative control scheme for *Hulk*. Rather than single button presses for Hulk's primary actions, punch and jump, he prototyped in GameMaker (and later implemented on the DS) a two-part system that used the X button as an amplifier for the punch and jump buttons.

We called it the Rage button, since it gave Hulk a surge of energy which amped his jump into a super one, and upgraded his punch into a doubly destructive blow. Our designer, the primary champion for the control scheme, felt this particular setup helped connect the user to Hulk's actions.

While understanding our intentions, our publisher had reservations with the Rage mechanic. Our publisher preferred that every player, even the youngest, have the ability to make Hulk as cool as possible. Accessibility is, as we all know, an important selling point.

We spent numerous calls and visits with the client deliberating the Rage button's merits and ferreting interpretations out of user tests.

After protracted debate (including internal debate), we peacefully concurred and removed the two-button combo to trigger Rage effects.

However, the debate over this feature unnecessarily affected several cycles, as well as the project's overall timeline.

The publisher's job is to know what will appeal to the broadest base of consumers, and while being too passive and "rolling over" to any suggestion is a recipe for gameplay disaster, so is a developer being too inflexible or too precious about their own design.

In hindsight, we should have resolved the debate much earlier by simply removing the Rage button and moving on. Once the decision to remove the Rage button was made, the issue was buried and our progress proved unhampered.

Conclusion

Ultimately, though the fully destructible environment was not as big a selling point as we had hoped, we are very pleased with *Incredible Hulk* for the Nintendo DS. Our team overcame some daunting challenges and learned many lessons that they have taken with them to their new projects.

Though we had some conflicts with the publisher, they were never highly contentious or irresolvable. In addition, our relationship improved greatly over the last several development cycles and in testing.

We finished our game on time and on a positive note. All things considered, the game has been received well by the public and seems to have many fans. Most importantly, however, the game is fun to play.

Return to the full version of this article