**Script**

Hi, I’m Jason Helms. **Along** with my traditional rhetoric classes, I teach classes in the broader DH umbrella, but closer to the new media studies side of things: **animation**, **web design**, **video game design**. Throughout these classes, I try to create a space for students’ passion, to teach design as creative problem solving. This makes the class enjoyable and meaningful for me and, I hope, for my students. What I realized **recently**, is this also sets them up to have their labor exploited after college, and perhaps in my very classes.

In this talk, I will offer a **theory** of students as laborers, **compare** student labor to video game development crunch, **explain** why we teach our students to crunch, and offer a few **suggestions** on how we can avoid exploiting student labor.

I want to claim **first** that student labor exists in our classes. That may seem an odd claim, but when we think of student labor, we tend to think of internships, work-study jobs, and extracurricular activities. Instead, the primary labor our students are doing is coursework. We’re happy to remind them that their job is being a student and being a student is a full-time job. There’s an oft-quoted rule of thumb that for every hour in class they should expect two hours of homework. A student taking the typical 15-hour load is actually working a 45-hour a week job.

Last semester I taught a web design class and a game design class. Both of these courses required a not insignificant amount of coding and few, if any, of the students had any experience in the languages I was teaching. So there’s a pretty big learning curve. Despite that, the students put in a great deal of work. Based on an informal poll, the students in my game design course reported that they put in two to three times as much work in my class as in any of their other classes. I’m not sure the course requires that, but making games is really fun and kind of **addictive**.

While I don’t teach STEM courses, I can speak a little to the difference between humanities courses and other majors. **STEM** courses often feature **specific** means and ends, so students know when they’ve done enough work. On the other hand, **courses** that utilize design-thinking methods emphasize problem solving and encourage creatively approaching assignments. In these courses, it’s not the **end** that is vague but the means. **Humanities** courses often have broader goals than stem courses. My **classes**, and many DH and DM classes, leave the **ends** and means vague, which is exciting and fun, but can lead to a higher workload for students.

Over the last fifteen **years**, there’s been a remarkable body of literature arguing for the **student**-as-co-producer model of education, rather than the **student**-as-consumer model.

This conversation began with a 2003 article **entitled** “Considering the Labor Contributions of Students: An Alternative to the Student-As-Consumer Metaphor.” It seems that the subtitle has gotten a lot more attention than the title. The authors’ proposed co-producer model is attractive because it puts responsibility back on to students—they are responsible for doing their own labor to become educated. Much of the subsequent articles have a celebratory quality: it’s no longer our fault if students don’t learn! As Allistair McCulloch points out, viewing students as consumers alienates them from their education, their teachers, and their classmates. Rather than considering student labor as possibly exploitative, scholars seem positively giddy in their readiness to exploit an argument about labor to absolve themselves of responsibility. They’re not wrong. In pedagogy, as everywhere else, agency is distributed across a network of actors. But today, I’d like to respond to the original title and actually consider student labor and the ways we might be exploiting **it**.

We don’t think of coursework as labor because students aren’t getting paid and we think of labor as inherently exploitable. But who benefits from the surplus labor of classroom work? Teachers do, through evaluations and teaching portfolios that showcase student work. The department and the university itself does because students who do well end up being our models that we use to advertise our programs. The current political economy benefits by creating laborers who are not just ready to be exploited but willing to joyfully exploit their own labor. I suspect some of this happens in every class. Cramming for a final or writing a 10-page paper in a week are signs of labor being exploited. Unfortunately, the typical faculty response to that exploitation is the expectation that students have poor time management skills that lead to end-of-semester cramming. Students drink, party, and go to football games. They’re not serious like us. And then we turn that evaluation into an expectation. We want students to spend weeks on those end-of-semester papers and projects, but at what point in the semester do we stop teaching new material so that they have a chance to? Instead, we assume they’ll leave it to the last minute, and that means we can keep adding new material right up until the last week of class.

There has already been admirable work done to show **how** rhetorics of openness and collaboration in DH mask the realities of student labor (Anderson, et al.). What I’m interested in today, though, is the **ways** the enjoyment of the labor itself masks its exploitation.

I want to draw a **connection** between the culture of cramming in college and the culture of **crunching** in video game development. I think what I have to say today applies to all teachers, but I think it’s especially true for those of us who teach students to make things with computers, and even more true when they make fun things. I’ll walk through some of my reasons for thinking that as I go on.

So, what is **crunch**? Crunch is overtime, spread across more than a week, typically associated with software, especially game, development.

For video game development, **crunch** is situated around big release dates, typically the fall so that games will be available for Christmas. That means the typical 40-60 hour workweek becomes 80-100 for the entire **summer**.

“Countless studios have undertaken crunch, sometimes extending to mandatory 80-100 hour work weeks for years at a time.” (Tozour)

Crunch is **widespread**, with a recent IGDA survey reporting that over half of all Game Development employees say that crunch is an expectation at their job. Diving further into the numbers is unsettling, with **about** a third of employees working 60-70 hours a week during crunch, and one out of seven working more than 70 hours a week during crunch (23). Even more **disturbing** is the fact that things are actually better than they used to be, with a general downward trend in crunch over the last fifteen years (33).

Crunch perpetuates itself through professionalization and **story** telling. “Surviving crunch is valorized and can be borne as a badge of honour and a signal of being a true developer. In the expressions of solidarity that are made through shared stories, the underlying message is one of gained experience and credibility and the sense that to truly belong is to have such a story to share” (Weststar 2015, 1244). This sounds a great deal like conversations I have with my students about the long hours I pulled in **school**. Those stories are meant to commiserate but also to inoculate and professionalize students into a system where labor practices are unfair and unproductive.

Meanwhile, the evidence is clear that crunch doesn’t **actually** work:

When used long-term, Crunch Mode slows development and creates more bugs when compared with 40-hour weeks.

More than a century of studies show that long-term useful worker output is maximized near a five-day, 40-hour workweek. Productivity drops immediately upon starting overtime and continues to drop until, at approximately eight 60-hour weeks, the total work done is the same as what would have been done in eight 40-hour **weeks**.

In the short term, working over 21 hours continuously is equivalent to being legally drunk. Longer periods of continuous work drastically reduce cognitive function and increase the chance of catastrophic error. In both the short- and long-term, reducing sleep hours as little as one hour nightly can result in a severe decrease in cognitive ability, sometimes without workers perceiving the decrease. (Robinson)

Academics don’t **typically** have too much crunch. Don’t get me wrong, we’re terrible at time management, regardless of how much we like to say that our students are worse. But, our due dates are often fairly negotiable and it’s the exception when we’re working at the last minute under a big deadline.

Our students, on the other hand, are faced with hard due dates every semester. If they don’t get something in on time, they fail. So, they develop time management coping mechanisms.

So, if crunch doesn’t work, **why** does it happen? For a lot of the same reasons our students do it. Writing for *Waypoint*, Tanya Short argues that crunch has ten self-replicating factors, that is, the reasons employees crunch not the reasons management asks them to. Rather than explain how they work in game development, I’ll just jump right to talking about how we teach students to recreate it for themselves. I’ve also adjusted her ten points into a few smaller points

**When** something is new and exciting, we tend to ignore its flaws. Often, we are introducing our students to innovative new methods. I think we’ve each seen students’ eyes light up when they realize all that they can do with the powerful tools we’re introducing them too. That light is not the cool reason of clarity. That joy obfuscates the fact that they continue to pour hour after hour into our projects at the expense of their health. If you couple this with the perfectionism that likely got them into our classes in the first place, it can get worse even after the honeymoon is **over**.

Often our students have a basic version of the **protestant** work ethic already coursing through their veins, in which work is its own reward and industriousness is a virtue. The flipside of this is guilt over their privilege of doing knowledge work. If you grew up thinking only manual labor counts, it feels shitty to complain about being tired from hours of coding. On the other hand, you can turn those hours into a badge of honor by bragging about how much they’ve sacrificed, even egging each other on by creating a dark camaraderie of suffering. Pekka Himanen’s **hacker** ethic seems like it might counter this destructive cycle. The hacker ethic privileges joy and creativity with a free-flowing work schedule. However, capitalism is very good at using this freer ethic to oppress workers, turning that joy into another mechanism for **exploitation**.

As we repeat this cycle students develop a culture of crunch and of learned helplessness. This is the way it’s always been, and I kind of like it, plus who am I to think I could change it. As we show other faculty the amazing work our students are doing, we end up creating a larger culture of crunch across academia.

Scholars have argued that playing games makes students more critical of technology and culture (Alexander, Frasca). The culture of crunch in video game development calls that into question. If playing games makes us critically aware, why are game developers so willing to offer themselves to be exploited? If we want our students to be critical users and makers of technology, clearly playing games isn’t **enough**.

One possible solution I’ve tried in my own classes is the **elimination** of grading. I’ve based my own system on Margaret Syverson’s learning record, used in many writing classrooms nation wide. In addition to the various assignments throughout the semester, students turn in self-evaluation letters at the beginning, middle, and end of the semester. The first letter asks them to evaluate themselves across five course-specific areas. The second and third letters ask them to evaluate how they’ve developed in those five areas over the course of the semester and use that evaluation to argue for a grade. I give them clear guidelines on how to grade their own work and extensive feedback on assignments throughout the semester that they can use to determine their grade.

**Such** a system better approximates the various ways they will be evaluated outside the university. While most employees are evaluated, those same employees often have to make arguments for raises and promotions. They also need to learn to quantify their labor. My students are never allowed to say “I deserve this grade because I worked hard.” Instead, they have to tell me the amount of hours they spent on an assignment if they want to make any arguments about effort.

My goal is to get students to think critically about their own labor and take ownership of their learning. My hope is that such efforts will transfer to other courses and beyond college. I am in no way trying to instill a new work ethic. Instead, I want my students to be critical of the work ethics they already exhibit.

In closing, I want to recognize how little I’ve offered and how much is yet to be done. It’s only in the last year that I’ve been thinking of my students as laborers. What effect does the fact that students aren’t being paid have on this argument? My hope is that this is just the beginning of a larger research project for myself and others, the goal of which is to teach students to resist their own exploitation and teach professors to resist exploiting student labor.

**Citations**

Alexander, Jonathan. "Gaming, Student Literacies, and the Composition Classroom: Some Possibilities for  
Transformation." *College Composition and Communication* 61.1 (2009): 35-63. Print.

Anderson, Katrina, et al. "Student Labour and Training in Digital Humanities." *Digital Humanities Quarterly* 10.1 (2016)Print.

Carey, Phillip. "Student Engagement and the Threat of Consumerism: Testing Assumptions." *International Journal of Advancement in Education and Social Sciences* 3.1 (2015): 1-7. Print.

Frasca, Gonzalo. "Videogames of the Oppressed: Critical Thinking, Education, Tolerance, and Other Trivial Issues." *First Person: New Media as Story, Performance, and Gam.* Eds. Noah Wardrip-Fruin and Pat Harrigan. Cambridge: MIT, 2004. 85-94. Print.

Halbesleben, Jonathon R. B., Jennifer A. H. Becker, and M. Ronald Buckley. "Considering the Labor Contributions of Students: An Alternative to the Student-as-Customer Metaphor." *Journal of Education for Business*.May/June (2003): 255-7. Print.

McCulloch, Alistair. "The Student as Co-Producer: Learning from Public Administration about the Student–University Relationship." *Studies in Higher Education* 34.2 (2009): 171-83. Print.

Robinson, Evan. "Why Crunch Modes Doesn't Work: Six Lessons." 2005 2005. Web. <[https://www.igda.org/page/crunchsixlessons](https://www.igda.org/page/crunchsixlessons" \t "_blank)>.

Schreier, Jason. "The Horrible World Of Video Game Crunch." May 15, 2015 2015. Web. <[https://kotaku.com/crunch-time-why-game-developers-work-such-insane-hours-1704744577](https://kotaku.com/crunch-time-why-game-developers-work-such-insane-hours-1704744577" \t "_blank)>.

Short, Tanya. "The Curious Appeal of Crunch." October 28, 2016 2016. Web. <[https://waypoint.vice.com/en\_us/article/yvwnqk/the-curious-appeal-of-crunch](https://waypoint.vice.com/en_us/article/yvwnqk/the-curious-appeal-of-crunch" \t "_blank)>.

Tozour, Paul. "The Game Outcomes Project, Part 4: Crunch Makes Games Worse." January 20, 2015 2015. Web. <[http://www.gamasutra.com/blogs/PaulTozour/20150120/234443/The\_Game\_Outcomes\_Project\_Part\_4\_Crunch\_Makes\_Games\_Worse.php](http://www.gamasutra.com/blogs/PaulTozour/20150120/234443/The_Game_Outcomes_Project_Part_4_Crunch_Makes_Games_Worse.php" \t "_blank)>.

Weststar, Johanna, Victoria O'Meara, and Marie-Josée Legault. *Developer Satisfaction Survey 2014. Summary Report.  
International Game Developer Association.* IGDA, 2018. Print. Summary Report .

Weststar, Johanna. "Understanding Video Game Developers as an Occupational Community." *Information, Communication & Society* 18.10 (2015): 1238-52. Print.