THE SINGAPORE-MIT GAMBIT GAME LAB PRESENTS

7 MYTHS OF VIDEO GAME DEVELOPMENT

TEXT BY MATTHEW WEISE
ILLUSTRATIONS BY MIKE RAPA
DESIGN BY GEOFFREY LONG
APOLOGIES TO SCOTT MCCLOUD
The Singapore-MIT GAMBIT Game Lab is a research and development lab devoted to the study of videogames. Every summer GAMBIT holds a game development workshop in which several student teams must take a game from initial concept to final polish in nine weeks. This gives GAMBIT staff the opportunity to observe and catalog common problems faced by first-time game makers.

These problems arise from common myths about game development many students believe before they even arrive at GAMBIT. Anticipating and deprogramming these myths are the key to successfully teaching students to be efficient game developers.

For more about us, check out our website at http://gambit.mit.edu!
1. TESTING IS NOT VERY IMPORTANT!

**Problem:** Many students learn the hard way that testing should be an integral part of any development process from day one. They neglect testing their builds, and then watch in horror as bugs ravage their project hours from a deliverable.

**Solution:** Rigorous testing is an integral part of GAMBIT. Each team has a dedicated test lead who is mentored by a testing supervisor on GAMBIT staff. The test lead must conduct full stability testing with every deliverable.
Problem: Some students, especially those with little or no programming experience, arrive with the idea that designers are the sole creative component of a game development team. They think their job is to tell others, especially programmers, what to do without much feedback or creative collaboration.

Solution: GAMBIT staff holds regular meetings with all team managers and discusses how to deal with existing or potential problematic personalities on each team. If necessary, team roles can be reassigned.
3. PROGRAMMERS DON'T HAVE TO DO WHAT ANYONE ELSE TELLS THEM!

Problem: Some programmers, especially those who are used to making games on their own, can develop an attitude of disinterest and/or hostility towards the collaborative process. This can be very destructive to a small project in which most or all coding responsibilities are in the hands of a single programmer.

Solution: Sometimes programmer roles can be reassigned, but it is harder to reassign a programmer who may have already written much of the game code. In especially difficult cases GAMBIT staff can reject builds or features that were created outside team collaboration.
Problem: Many students scope a single-player project and then plan on adding multi-player as a feature, not taking into account that the networking tech and testing coordination of even a small multi-player feature upscales a project exponentially.

Solution: We explain the scoping consequences of multi-player before development begins and advise students not to treat multi-player as a minor feature.

However, if students insist on attempting to scope a multi-player project we don’t stop them. We instead let them deal with their project spinning out of control for the first few milestones, meeting with them regularly to discuss their scoping difficulties. This helps students arrive at their own conclusion that multi-player development is radically different than single-player development.
5. Overworking Yourself is A-OKAY!

**Problem:** It is difficult to get some students to stop working. Rather than pace themselves and manage their time effectively, they prefer not to plan and just work until they get too tired to continue. This often disrupts the schedules (and affects the morale) of other team members, since dependencies force the team to conform to the schedule of an erratic artist or coder.

**Solution:** GAMBIT enforces strict work hours as much as possible. We do not allow students to take work home, and we make sure they are out of the office on time each day.
6. GUI is just something you slap on at the end!

Problem: First-time game developers often underestimate the importance of user interface design, specifically in regards to visual feedback. Teams can spend their entire development cycle implementing and tweaking gameplay while spending little or no time on designing and testing a good graphical user interface.

Solution: At GAMBIT we take user interface testing as seriously as stability testing. Teams must conduct user-testing in the first few weeks, and user feedback about user interface issues must be incorporated into subsequent builds. GAMBIT staff instructs teams on how to properly conduct user testing, such as how to avoid giving too much information to a user during a testing session.
Problem: Many students expect their game to be fun at first iteration, and are disappointed – even depressed – when it is not. It is sometimes hard for students to understand that a fun user experience can often only be created through much iteration and testing.

Solution: GAMBIT students are taught from the beginning that iteration is the cornerstone of game development. All teams must begin with paper prototypes before coding even begins, and design must be re-evaluated after every user-tested deliverable.