TOUCHE: Coding Competition Software

Daniel Sanders, Tyler Garcia, Caleb Stevenson, Matt Goldsberry, George Harwood, Alexander Wagner

Department of Computer Science and Engineering, Taylor University, Upland, Indiana 46989

Introduction

This project has been in development for many years now. It was originally written in the late 90's but was last updated in 2005. It's written in mostly PHP and HTML, with no CSS or modern design tools. At its core it is web based software that runs coding competitions and allows for all the functionality you would expect during a competition. Just to hit on the basics of what a coding competition is, during a contest teams have a list of problems to work on and they attempt to write code to resolve the problems. They then submit their code and the touche system will test it to see if it gives the correct output. The current touche code first allowed an administrator to create a new contest and setup and configure the problems and other settings. Then once a judge starts the contest teams are allowed to see the problems, work on their code and then submit for judging. The system checked the code and provided the judges with a recommended response, which they could accept or change. It was then able to give scores and rank the teams by the number of problems they had gotten correct. All in all it was a working system, that had been used for many competition since it's inception. But we were tasked with making it better, and turning it into a modern system with many added features.

Approach

As we started off this project we had many different directions to go. The basis for the improvements we needed to make came from some feedback we had received. The code had been lended out to a different university to host their own contest. After running their contest they provided Taylor with a large list of potential bugs, questions and improvements. We started with this list, the base code and went from there. We began by gathering what requirements we could about which potential features were high priority. The key thing that made this project different was the fact that the current code was a working version. We needed to keep in mind that if at the end of the day our new code lacked features or contained critical bugs then there was no way it could be utilized. We also had an overarching goal (no matter how many features we were able to add to the code) of converting this software to a live CD format. The future vision for this setup is that someone could be given a server CD and a user CD. They put the server disk into a machine and they are automatically up and running the contest framework. Then the user disk is inserted into the machines teams will be using and they are ready for the competition.

Design

Since the software was developed in the late 90's, there were not very many options for stylizing the software. It was built without CSS, or any kind of styling language. That being the case, we started off by: completely redesigning the look and feel, branding, changing the color scheme, and implementing Twitter Bootstrap.

Take a look at the new logo:

This is the color scheme we decided on:

By the end of the project we had gone through the process of creating prototypes using illustrator, and changing over fifty PHP files to correspond with this new look.

Development

The development process of Touche went through a variety of steps, using Pivotal Tracker to keep everyone informed of everyone else's progress. The first thing that happened in the development process was setting up the server so that we could see what we were changing. After getting the server up and running, which took a decent amount of time, we started to look into updating a lot of the functionality that was either bugged or missing completely. One of the important aspects of what we had to do was to revamp the structure of what admins could do to make the experience more user-friendly. With that in mind, we moved things around in the admin panel and even moved the "Start Contest" page from the judge panel to the admin panel. A lot of time was spent fixing bugs that seemed to pop up out of nowhere, but in the end we had a product that was both cleaner and more functional than the original product.

Added Features

- Updated the user interface to improve visuals, simplify workflow, and solve broken interfaces that were present with the original design.
- Created live CDs for both the client terminals and Touche server, reducing setup time for programming competitions from hours to minutes, with no need for expertise.
- Added a test competition functionality, which allows administrators to check if the server is properly configured before the competition begins.
- Enhanced Java judging stability, improving upon the frequent false positives and crashes in the old system.
- Changed how competitions are begun so that now the administrator is the only one who can start.
- Improved Touche's clock, allowing for better coordination from multiple locations, as well as improving user experience.
- Numerous other bug fixes affecting all levels of Touche's functionality.

Challenges

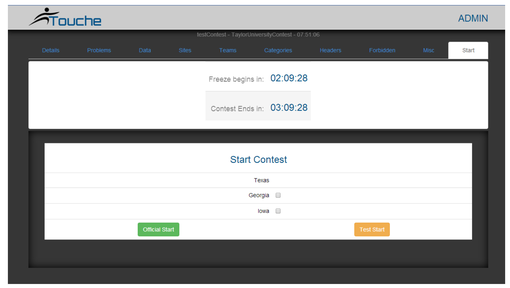
- Making the server run properly was a challenge that we started off with needing to solve. The combination of programs that we needed to use simultaneously created bugs that were difficult to remove.
- Operating across multiple accounts and needing to re-sign in to each account as soon as we accessed a different account made testing take more time and more frustrating than it needed to be.
- With having multiple people working on the same project, combining our code into one cohesive piece was a challenge that added many bugs to the system. Using GitHub, which we all have limited experience on, did not help either.
- As none of us had learned PHP at this point, learning a new language on top of coding and fixing bugs was difficult.
- Access to the various pages we needed to automatically view was often locked due to permissions on the files, meaning we had to spend time manually changing permissions to those files before automating the process.

Future Improvements

A developer's job is never done. Throughout our time with the software we made a lot of improvements, but did not get to everything we wanted to. There were two fairly high priority things that we were not able to achieve. The first was that we wanted to allow teams access to the compiler. With the existing system the teams get counted off if their code does not compile. Obviously they would not submit code that they did not think would compile. It does not seem fair to penalize them for differences between their compiler and the contest compiler. The other big thing would have been adding python support for teams. Currently the system only supports C, C++, and Java. There were also smaller things that would have been nice to be able to do. One of them is a bug that we simply could not find. The bug has broken the functionality that allows the admin to specify whether the judges are allowed to see the team name of the team that they are judging. This would be useful to prevent prejudice among judges.

Conclusions

This code was very outdated when we received it and was not designed very well. We hope that we left the current version of Touche much more up to date and easier for future improvements to be made. We corrected a good number of errors that were known bugs and some that we found on our own. There are still many bugs in the system that it would be nice to get dealt with to improve the product even more. We hope that the inclusion of a live CD will allow for more users to have access to this system to promote coding competitions. Touche is still a work in progress, but we know that we have improved a system that was barely working into a system that runs much more smoothly and has the ability to become a very good system in computer programming competitions.



For More Information contact Dr. Geisler.