Randomized Content

High level aspects of adding randomized content to your assignments

Methods of Randomization

There are two main routes one could take to implement randomization, LaTeX and Sage.

LaTeX randomization - works in theory, but not in practice.

LaTeX is a Turing-Complete language, which means it certainly can provide randomization. Indeed, I have actually written some useful randomization packages for LaTeX myself.

Unfortunately, although one could implement this randomization into the LaTeX code for Xronos, the problem comes in how Xronos is currently published[1].

Currently, Xronos is published using the Xake tool, which essentially translates the TeX files into html and javascript (among other things), and then pushes those files to the server. But this means that all the LaTeX content is run and translated into the webpage *when the page is published*. In other words, the only time LaTeX is run (and thus the only time any randomization at the LaTeX level that occurs) happens when an update is published to the web.

So, in order to get any kind of LaTeX-level randomization, you'd need to republish the assignment each time you wanted to randomize it. Moreover, this would only randomize the content served to the server, not to the students - in other words, every student would necessarily get the *same* randomized content, it wouldn't be different for each student in any way.

Sage randomization

The other option is to use the sagetex package for LaTeX and sage code to implement randomization. This has the benefit of having a much more powerful programming syntax and tools to enable much more complex mathematical manipulation. In order to use sage however, you would likely want to get a local installation of sage and sagetex in order to be able to debug content locally before

Learning outcomes:

[1] As an important note, this will change in the next version of Xronos - which will compile LaTeX directly in the browser, which opens up all kinds of interesting possibilities.

publishing it to the web. You can read more about getting a local installation of sage and sagetex here - the details are beyond the scope of this document.

In the case of sage, the seed for all randomization is set behind the scenes at the server level. This is important for three reasons.

(a) The randomization needs to be stable so that each time a given student loads a page, they get the same problem - rather than getting new problems every time they see the page (this would be especially confusing when looking back at problems they got "right", since the displayed question and the provided answer wouldn't match up once the displayed problem randomized to something new).

(b) A design decision was made to allow students to get the same problems in the same order for a given randomization for a given semester. This allows them to work together on problems, since they would all get the same variation of a problem together, then they could all hit "try another" and end up with the same problem on the next iteration as well.

(c) Instructors can load a page and see the same problems in the same order, that students got, since they are also loading the same content with the same randomization seed.