# Simulador de Scoreboard

Resumen del Proyecto

Pablo Zimmermann

8 de marzo de 2016

# 1. Descripción del Proyecto

- **¿Qué hace?**

El proyecto es un "Simulador de Competencias de Programación en tiempo real". El simulador, a partir de los resultados de alguna competencia pasada, brinda exactamente la misma información que recibe un equipo durante una competencia oficial, es decir, un "scoreboard" que le informa a los programadores entre otras cosas que posición ocupan y que problemas están realizando los competidores rivales.

- **¿Por qué?**

En mi breve experiencia como competidor de competencias, hemos "simulado" muchas competencias pasadas (más de 70) a modo de entrenamiento. Sin embargo a lo largo de estas simulaciones hemos notado que es extremadamente distinto competir "a ciegas", es decir, solo teniendo la lista de problemas a resolver, que competir en una competencia real. Esto es no sólo por los nervios que ello contrae sino más bien por el hecho de que en una competencia real uno tiene muchos equipos rivales que resuelven problemas al mismo tiempo que uno y por lo tanto en todas las competencias uno posee exactamente la información de que problemas están intentando los otros equipos, en que minuto resolvieron un problema, y cuanto intentos hicieron.
Esta información es esencial a la hora de seleccionar que problema resolver. En toda competencia uno generalmente tiene tiempo para resolver, en promedio, menos de la mitad de los problemas. Si a esto le sumamos que los problemas están pensados para tener un nivel de dificultad muy variado (cosa que todos los equipos puedan resolver alguno y nadie resuelva todos), y aparte no están ordenados por orden de dificultad queda claro que la información sobre que problema están realizando los otros equipos es una información muy útil en la elección de que problema resolver.
Por lo tanto el objetivo de este proyecto es brindar una herramienta a todos los competidores de este tipo de competencias, con el objetivo de brindarles la misma información que van a tener en una competencia y que así puedan practicar no sólo la resolución de problemas sino que táctica deben usar y como se tienen que organizar como equipo de cara a lo que les queda de competencia y los problemas que aún no pudieron resolver.
El proyecto se basa en un servidor creado con la herramienta Snap (Inserte referencia aquí) que provee al usuario funcionalidades para cargar y simular competencias pasadas.

# 2. Manual de Uso e Instalación del Software

- Bajar y descomprimir el programa.
- Instalar Cabal (sudo apt-get install cabal).
- Instalar snap. (cabal install snap).
- Instalar otro paquete necesario. (cabal install x).
- Compilar y ejecutar el programa. (ghc Main.hs)
- Dirigirse a la página del servidor. (http://0.0.0.0:8000/)

- Crear un usuario y loguearse.
- Cargar el simulacro (ya sea por un archivo de Html o un link).
- Elegir el simulacro ya cargado.
- Setear opciones antes de arrancar.
- Apretar Start y leer el contest vía la pestaña Score.

# 3. Modulos y Archivos adicionales

## 3.1. Modulos

Estos son los modulos que posee el simulador:

- **Parser**

Parsea un scoreboard de una competencia pasada, y transforma la información en un formato deseable, para luego ser trabajada.
Provee solo la función teamsParseados, que dado un texto en formato HTML devuelve la información de la competencia para ser usada.

- **Parser Kattis**

Ante los múltiples formatos de los scoreboards, se necesita un parser particular para cada tipo. A modo de ejemplo, se provee el parser utilizado para leer los scoreboards utilizados en las competencias mundiales (que lo vamos a llamar el "formato Kattis"). Para poder leer otros scoreboard generalmente se necesitan cambios menores (modificar algunas constantes y en algún que otro caso alguna otra función minimamente). El módulo usa una librería de HTML llamada HXT para parsear el código HTML.
Como aclaración sobre este modulo, esto fue lo primero que realizé (hace casi 1 año) muy rudimentariamente y es una parte que yo considero como menor dentro del trabajo completo. El primer cambio a futuro que considero hacer es trabajar sobre este módulo, haciéndolo todavía más reusable y refórmandolo por completo (usa un archivo externo en lugar de una mejor mónada). Queda como trabajo pendiente.

- **Score**

Este modulo se encarga de la elección del scoreboard a simular. Es la interacción entre el usuario y el modulo Parser.
Provee 3 funciones básicas, cargar un scoreboard desde un archivo, cargar un scoreboard desde internet y seleccionar un scoreboard ya cargado para ser usado en la simulación.

- **Teams**

Es el módulo lógico que trabaja con los equipos del simulacro. Provee una única función que dada la información de los equipos y un minuto del simulacro, cálcula la información que debe ser mostrada en ese minuto, ocultando la información futura. Lo que devuelve es la tabla ordenada de los equipos en ese minuto lista para ser mostrada.

- **HtmlContestPrinter**

Se encarga de generar el html del scoreboard, en base a la tabla actual de los equipos.
Todas las funciones auxiliares generan Strings, que son usadas por generate para finalmente imprimir el scoreboard completo.
Por razones de simplicidad y de generar un scoreboard igual al que proveen en el mundial de programación, se utilizan dos archivos que tienen las directivas usadas para lograr el mismo formato.

- **User**

Este modulo se encargar de ofrecer funcionalidades extras al usuario, a fin de brindarle más opciones de cómo desea simular.
Provee funciones para agregar nuevos equipos, borrarlos, agregar nuevos envíos de los equipos, listar todos tus equipos, cambiar la velocidad de simulación, arrancar el simulacro, pausarlo, reanudarlo y reiniciarlo.
Para hacer esto, trabaja modificando un estado general del contest (ContestState). Este estado es parte de la mónada general en forma de referencia y por lo tanto se usan funciones de IORef para modificarlo o accederlo.

- **Application**

Este módulo simple es creado automáticamente al utilizar el comando snap init.
Define el estado de aplicación y un alias para la mónada Handler, la cuál es la monada utilizada por el servidor.

- **Site**

Este módulo también es creado automáticamente al utilizar el comando snap init.
Es el modulo central del servidor. En él se definen todas las rutas y los handlers usadas en el servidor o sitio. Solo exporta la función .ªpp"que inicializa y combina todas las cosas juntas.
En sí es el mismo modulo creado automáticamente agregando las rutas definidas en el modulo Form.

- **Form**

Este módulo le provee al modulo Site todos los formularios para las distintas acciones que puede hacer el usuario.
Es la interacción entre lo que realiza el usuario y los modulos Score y User que realizan las funciones básicas de la simulación.
Usa una única estructura, la cuál usa según la cantidad de información que debe brindar el usuario para cada función.

- **Tipes**

Tipos y sinónimos de tipos que son usados a lo largo del simulador.

- **Auxiliar**

unciones auxiliares útiles que son usadas a lo largo del simulador.

- **Main**

Este es otro modulo creado automáticamente al utilizar el comando snap init. Provee funcionalidades que no fueron necesarias para los objetivos de este trabajo. De hecho no fue modificado absolutamente en nada, y solamente áctua de módulo general que iniciliza el sitio interáctuando solamente con el modulo Site.

## 3.2. Archivos Adicionales

En la carpeta /Html se encuentra todos los archivos con formato html que desean ser cargados. Una vez cargados, pueden ser borrados.

En /Internos se encuentran archivos internos con los que trabaja el scoreboard para simplificación. Se encuentran los archivos pre.txt y pos.txt que sirven para generar el scoreboard con el formato Kattis y un archivo in.html que ya fue mencionado.

En /log se encuentran historiales de trabajo. Son creados automáticamente con el comando snap init y fueron dejados ahí aunque no son usados.

En /Scores se encuentran todos los scores que ya cargados.

En /snaplets se encuentran todos los snaplets usados. Es otra carpeta creada de forma autómatica y actualmente solo contiene los templates usados para facilidad de interacción con el usuario.