

Proyecto Final de Procesadores de Lenguajes: Diseño e implementación de un Lenguaje de Dominio Específico (DSL).

Autores:

Miguel Pérez Bello
Tinguaro Cubas Saiz
Guillermo Rodríguez Pardo

Índice:

1. Descripción del proyecto	
1.1. Descripción del documento	
1.2. Descripción del proyecto	
1.2.1. Objetivo	
2. Especificaciones del diseño	
2.1. Descripción de la estructura del proyecto	
2.2. Descripción de los ficheros utilizados	
3. Palabras reservadas	
3.1. Listado de las palabras reservadas	
3.2. Ejemplo de uso de las palabras reservadas	
4. Especificación de la gramática	
4.1. Nociones básicas necesarias	
4.2. Gramática asociada al DSL	
5. Detalles de la implementación en JISON	
5.1. Ejemplo simple de una gramática básica en JISON	
5.2. Gramática asociada al DSL en JISON	
5.3. Consideraciones sobre la gramática	
6. Descripción de la página que aloja el proyecto	
7. Coordinación del proyecto	
7.1. Uso de repositorio Mercuria (Hg) para la organización	
7.2. Uso de repositorio Git Hub para la organización	
7.3. Software de libre distribución	
7.4. ¿Cómo obtener los códigos fuentes del proyecto?	
8. Primer prototipo	
9. Referencias	

1. Descripción del proyecto.

1.1. Descripción del documento.

El presente documento tiene por objetivo albergar la descripción formal del proyecto final realizado para la asignatura Procesadores de Lenguajes del itinerario de Computación del Grado en Ingeniería Informática (3º curso).

El documento describe las herramientas y pasos a seguir para comprender la estructura y el funcionamiento del proyecto, objeto de estudio.

1.2. Descripción del proyecto.

1.2.1. Objetivo:

Un Lenguaje de Dominio Específico (o DSL) es una herramienta que permite al usuario realizar ciertas tareas, comúnmente laboriosas, con cierta facilidad. La ventaja principal de este tipo de lenguajes es que las acciones a realizar están asociadas a ciertas palabras propias del lenguaje, sin que éste sea tan complejo como un lenguaje de programación convencional (Pascal, C, C++, Ruby, Java, ...).

Ejemplos de tareas que pueden ser llevadas a cabo con un DSL podrían ser:

- La creación de recetarios de cocina, tan sólo usando las palabras reservadas “receta=N”, “ingrediente=nombre”, “cantidad=C”, etc, ...
- El diseño de test o exámenes tan sólo especificando la asignatura, redactando las preguntas (con sus respectivas respuestas), indicando las repuestas correctas, detallando el método de puntuación, etc ...

El objetivo principal del proyecto consiste en la creación de un DSL para facilitar a las personas que practiquen algún tipo de deporte, la creación de su tabla semanal de entrenamiento. Este objetivo abarca poder especificar al máximo la información necesaria para organizar de la mejor manera posible la tabla de entrenamientos semanales de cualquier tipo de deportista, tanto de un futbolista, como de un aficionado al culturismo.

La información que se pretende manejar es aquella relativa a estructurar los ejercicios semanales que debe realizar el deportista en cuestión, tal como el número de la tabla de entrenamiento, una descripción (opcional) del objetivo del entrenamiento de esa semana, el día de la semana, la actividad en cuestión, la hora de comienzo, la duración de la actividad y cualquier tipo de nota al respecto.

Proyecto Final: diseño de un DSL.

2. Especificaciones del diseño.

Las herramientas utilizadas en el desarrollo de este proyecto se basan en el uso del metalenguaje interpretado HTML para albergar la interfaz gráfica de usuario (o GUI), cuyo aspecto visual se define usando hojas de estilo CSS. Se utilizará el lenguaje JavaScript para desarrollar el analizador (léxico y semántico) del lenguaje, y nos apoyaremos en un módulo de JACK (JSON) para la descripción de la gramática (G) que genera el lenguaje (L(G)) cuyo dominio se especifica en el apartado anterior.

2.1. Descripción de la estructura del proyecto.

La estructura de ficheros y directorios se define atendiendo a los estándares utilizados a lo largo del transcurso de la parte práctica de la asignatura a la que este proyecto debe su motivación. Esta estructura consiste en una serie de directorios principales en los que se separan los ficheros del proyecto atendiendo al formato y a la función que desempeña cada uno de ellos. Los directorios que conforman la implementación del proyecto son:

- “Estilo-CSS”: carpeta que aloja las hojas de estilo necesarias para especificar el estilo de la GUI. Contiene el fichero “global.css”.
- “JavaScript”: contiene los ficheros escritos en el lenguaje JavaScript.
- Libs: Es el directorio donde se incluye la librería “jquery.js”, que será utilizada en el desarrollo.
- El directorio principal del proyecto contiene el “index.html” y el fichero “RakeFile” de compilación del proyecto.

2.2. Descripción de los ficheros utilizados.

Los ficheros de los que se compone el proyecto son los que se especifican a continuación:

- “Estilo-CSS/global.css”: comúnmente denominado “hoja de estilo”. Sirve para especificar el estilo del formato con el que se visualizarán los componentes de la GUI del proyecto.
- “JavaScript/main.js”: crea las instancias y ejecuta el flujo normal del proyecto.
- “JavaScript/parse.js”: contiene la implementación en código JavaScript del parser que valida y “tokeniza” la entrada.
- “JavaScript/token.js”: contiene la descripción de los tokens que se obtienen de la entrada y permite ejecutar el análisis de la misma.
- “Libs/jquery-1.9.1.js”: versión 1.9.1 de la librería “Jquery”, necesaria en la implementación del proyecto.
- “Libs/underscore.js”: librería “Underscore”, necesaria en el proyecto.
- “index.html”: contiene la página principal en la que se aloja el proyecto.
- “Rakefile”: fichero para la compilación de los módulos que componen el proyecto.

Proyecto Final: diseño de un DSL.

3. Palabras reservadas.

Las palabras reservadas del DSL desarrollado permiten describir los ejercicios de las tablas de entrenamiento de una forma simple y específica. El dominio de nuestro lenguaje está muy restringido, puesto que lo conforman el conjunto de mnemónicos asociados a la especificación de una tabla de entrenamientos semanal, que a su vez está limitado por las palabras que permiten describir los ejercicios que se realizarán diariamente en la misma.

3.1. Listado de las palabras reservadas.

A continuación se detalla un listado de los identificadores utilizados para describir cada uno de los entrenamientos en las tablas semanales, o comúnmente denominados “palabras reservadas”:

Palabra Reservada	Significado	Sintaxis
Entrenamiento	Identifica el número de la tabla de entrenamientos que se va a crear.	Entrenamiento = Numero []
Día de la semana: Lunes, Martes, Miércoles, ..., Domingo.	Sirve para especificar el día de la semana en que se realizará el ejercicio que se pretende detallar.	+Día: (sustituyendo “Día” por el día de la semana correspondiente)
Ejercicio	Permite detallar el ejercicio a realizar, o un alias que permite recordar el mismo.	- ejercicio = “Descripción del ejercicio”
Hora	Especifica la hora de comienzo del ejercicio.	hora = número (hora en formato hora.minutos)
Tiempo	Especifica la duración del ejercicio.	tiempo = número (tiempo en formato número.minutos)
Fin del Entrenamiento	Especifica el fin de la descripción de la tabla de entrenamientos.	[] Fin del Entrenamiento.
[] (entre “Entrenamiento” y “Fin del Entrenamiento.”)	Encierra el cuerpo de la descripción de las tablas.	[...ENTRENAMIENTO...]

Tabla 1.1: listado de las palabras reservadas del DSL.

3.2. Consideraciones sobre las palabras reservadas.

La redacción de la tabla de entrenamientos semanales tiene algunas limitaciones en cuanto a la definición de las mismas, dichas restricciones atienden a cómo deben escribirse las palabras reservadas del lenguaje (ver sección “Sintaxis” de la tabla 1.1).

- Las principales limitaciones son:

1º- Las únicas palabras que componen el DSL son las citadas en la tabla 1.1.

2º- Las palabras reservadas deben escribirse tal y como se detalla en la tabla 1.1.

3º- El parser diseñado para el DSL, objeto de desarrollo, es case sensitive (sensible a minúsculas y mayúsculas), por lo que se recomienda prestar atención.

4º- El único ámbito presente en la descripción de una tabla de entrenamientos es el relativo a los corchetes “[]” (encierran el **body** de las tablas) que se escriben entre la cabecera “Entrenamiento=N” y “Fin del Entrenamiento.”, por lo que es una ventaja que facilita aprenderse la sintaxis del lenguaje.

5º- Los caracteres “+” y “-” indican el día de la semana y la descripción de un ejercicio, respectivamente.

6º- Los “datos” hora y tiempo van tras la descripción de cada actividad, separados entre comas simples (hora = x, tiempo = t).

7º- Por razones de no producir un innecesario oscurecimiento de la gramática y del propio código, no se utilizan las tildes del español (á, é, í, ó, ú).

3.3. Ejemplo de uso de las palabras reservadas.

Entrenamiento [

+ Lunes:

- ejercicio = “Correr”, hora = 20.00, tiempo = 60m;

+ Martes:

- ejercicio = “Correr”, hora = 20.00, tiempo = 60m;

+ Viernes:

- ejercicio = “Correr”, hora = 20.00, tiempo = 60m;

- ejercicio = “Hacer flexiones”, hora = 10.00, tiempo = 30m;

] Fin del Entrenamiento.

Proyecto Final: diseño de un DSL.

4. Especificación de la gramática.

4.1. Nociones básicas necesarias.

- **Gramática:** Una gramática G es una cuaterna $G = \{ \Sigma, V, P, S \}$, donde:
 - Σ : es el conjunto de símbolos terminales (caracteres con los que se diseñan las palabras reservadas).
 - V : es un conjunto (disjunto de Σ) que se denomina conjunto de variables sintácticas o categorías gramaticales.
 - P : es un conjunto de pares $\langle V, (V \cup \Sigma)^* \rangle$. Un elemento de P se denomina “producción”.
 - S : es un símbolo del conjunto V que se denomina “símbolo de arranque”.
- Nota: en “ $\langle V, (V \cup \Sigma)^* \rangle$ ” se utiliza el carácter “*” para referirse al “cierre de Kleene”.
El cierre de Kleene de un conjunto A (A^*), se define como:

$U(n=0, \infty) A^n$, donde $U(n=0, \infty)$ es el unitario desde $n=0$ hasta infinito, de todos los elementos del conjunto A .

- **Lenguaje generado por una gramática:** dada una gramática G , se denota por $L(G)$ al lenguaje generado por la misma, o:

$L(G) = \{ x \in \Sigma^* : S \Rightarrow^* x \}$, o sea, el lenguaje generado por la gramática G está formado por las cadenas que, partiendo desde el símbolo de arranque de la gramática (S), se puede derivar en cero o más pasos a las mismas.

4.2 Gramática asociada al DSL.

```
Entrenamiento → 'Entrenamiento' Cuerpo_Entrenamiento 'Fin'
Cuerpo_Entrenamiento → '[' lista_días ']'
lista_dias → Dia | Dia lista_dias
Dia → '+' 'Tdia' ':' Lista_descripcion_ejercicio
Lista_descripcion_ejercicio → Descripcion_ejercicio | Descripcion_ejercicio |
Lista_descripcion_ejercicio.
Descripcion_ejercicio → Nombre ',' Hora ',' Tiempo ';'
Nombre → '-' Nejercicio '=' ID_Nombre
Nejercicio → 'Tejercicio'
ID_Nombre → 'TvalorEjercicio'
Hora → NHora '=' ID_Hora
Nhora → 'Thora'
ID_Hora → 'TvalorHora'
Tiempo → Ntiempo '=' ID_Tiempo
Ntiempo → 'Ttiempo'
ID_Tiempo → 'TvalorTiempo'
```

Proyecto Final: diseño de un DSL.

5. Detalles de la implementación en JISON.

5.1 Ejemplo simple de una gramática básica en JISON.

+ Gramática:

```
% %  
S      : A  
      ;  
  
A      : /* vacío */  
      | A x  
      ;
```

+ basic_lex.jison:

```
% lex  
% %  
\s+ { /* quitar espacios en blanco */ }  
[a-zA-Z_]\w* { return 'x'; }  
  
/lex  
% %  
S      : A { return $1 + “ identificadores”; }  
      ;  
  
A      : /* vacío */ {  
          console.log (“starting”);  
          $$ = 0;  
        }  
      | A x {  
          $$ = $1 + 1;  
          console.log ($$);  
        }  
      ;
```


5.2. Gramática asociada al DSL en JISON.

```
%token TDIA TEJERCICIO THORA TTIEMPO TVALOREJERCICIO TVALORHORA TVALORTIEMPO
TVALORENTRENAMIENTO TVALORFIN
```

```
%lex
%%
```

```
\s+    { /*Eliminar espacios en blanco*/ }
```

```
Entrenamiento { return 'TVALORENTRENAMIENTO'; }    /*"Entrenamiento"*/
Fin           { return 'TVALORFIN'; }              /*"Fin"*/
```

```
[+]\s*(Lunes|Martes|Miercoles|Jueves|Viernes|Sabado|Domingo)\s*[:] { return 'TDIA'; }
```

```
Ejercicio     { return 'TEJERCICIO'; }
Hora          { return 'THORA'; }
Tiempo        { return 'TTIEMPO'; }
```

```
[a-zA-Z0-9_]\w* { return 'TVALOREJERCICIO'; }
[0-9]?[0-9][:][0-9][0-9] { return 'TVALORHORA'; }
\d+[m|M] { return 'TVALORTIEMPO'; }
```

```
"[" { return '['; }
"]" { return ']'; }
"+" { return '+'; }
'-' { return '-'; }
'=' { return '='; }
```

```
. { return 'TERROR'; }
```

```
/lex
%%
```

```
ENTRENAMIENTO: 'TVALORENTRENAMIENTO' CUERPO_ENTRENAMIENTO 'TVALORFIN' ;
CUERPO_ENTRENAMIENTO: '[' LISTA_DIAS ']';
LISTA_DIAS: DIA { $$push ($1); }
            | DIA LISTA_DIAS ;
DIA: '+' 'TDIA' '-' LISTA_DESCRIPCION_EJERCICIO ;
LISTA_DESCRIPCION_EJERCICIO: DESCRIPCION_EJERCICIO { $$push ($1); }
            | DESCRIPCION_EJERCICIO LISTA_DESCRIPCION_EJERCICIO ;
DESCRIPCION_EJERCICIO: NOMBRE ',' HORA ',' TIEMPO ',' ;
NOMBRE: '-' NEJERCICIO '=' ID_NOMBRE ;
NEJERCICIO: 'TEJERCICIO' ;
ID_NOMBRE: 'TVALOREJERCICIO' ;
HORA: NHORA '=' ID_HORA ;
NHORA: 'THORA' ;
ID_HORA: 'TVALORHORA' ;
TIEMPO: NTIEMPO '=' ID_TIEMPO ;
NTIEMPO: 'TTIEMPO' ;
ID_TIEMPO: 'TVALORTIEMPO' ;
%%
```

Proyecto Final: diseño de un DSL.

5.3. Consideraciones sobre la gramática.

La gramática propuesta como respuesta a la creación del DSL realizado, no presenta ambigüedad, por lo que no nos encontraremos ningún tipo de conflicto a la hora de implementarlo en JISON. Además, la simplicidad que tiene la gramática la hace disyunta del conjunto de gramáticas recursivas.

6. Descripción de la página que aloja el proyecto.

La Interfaz Gráfica de Usuario (GUI) de la aplicación, que aloja toda la funcionalidad que abarca el intérprete del Lenguaje de Dominio Específico diseñado para componer las tablas de entrenamientos semanales, está alojada en una página web escrita en HTML (ver fichero index.html del proyecto).

La GUI de la aplicación citada provee un área de texto en el que el usuario puede introducir el código que necesite para generar sus propias tablas de entrenamientos semanales, un botón que lanza un evento cuando es pulsado, evento que es capturado y que ocasiona la ejecución del parseo de la entrada, y la ejecución de las acciones semánticas a cada una de las instrucciones introducidas.

Además, la interfaz contiene una opción que permite visualizar un tutorial simple de uso del DSL; dicha información se muestra en una pestaña aparte en el navegador.

7. Coordinación del proyecto.

Para la coordinación del proyecto, nos hemos apoyado en las plataformas ó sistemas de control de versiones Git Hub y Mercuria, que permiten la creación de repositorios remotos en los que trabajar de forma controlada a través del uso de diversas ramas (o branches), en este caso dos por cada miembro del equipo de desarrollo software del proyecto.

El empleo de este modo de organizarse permitió separar las etapas del desarrollo sin que se produjeran conflictos entre versiones. Al tener delegadas las responsabilidades en el desarrollo, cada miembro del equipo pudo ocuparse de desarrollar una parte del proyecto para su final combinación (o merge).

Otra ventaja que resultó del uso de dichas plataformas fue la posibilidad de separar las etapas inherentes al desarrollo de las etapas propias de la fase de validación, de este modo cada miembro del equipo pudo realizar las pruebas de estrés al código que el mismo redactó, consiguiendo así mejores resultados.

A continuación podemos ver una pequeña descripción de los sistemas de control de versiones (SCV) anteriormente citados.

Cabe añadir, que el SCV utilizado mayoritariamente en el desarrollo del proyecto fue Mercuria, pero por razones de costumbre al realizar las prácticas de la asignatura, se optó también en replicar el repositorio a GitHub, que es donde se encuentra la libre distribución del mismo.

Proyecto Final: diseño de un DSL.

7.1. Uso de repositorio Mercuria (Hg) para la organización.

Debido a la gran importancia que se le ha impuesto al uso de sistemas de control de versiones, y a la gran mención actual de la plataforma Mercuria, como uno de los SCV más utilizados, se optó por delegar la responsabilidad de la organización del proyecto en dicha plataforma. Mercuria permite la creación de repositorios tanto públicos como privados, pero defendiendo la política de libre distribución del software, se optó por alojarlo en un repositorio abierto al público.

7.2. Uso de repositorio Git Hub para la organización.

Al contrario que en Mercuria, este SCV sólo permite la creación de repositorios públicos sin efectuar pago alguno (en la mayoría de casos).

Como se comentó anteriormente en gitHub podemos encontrar los ficheros fuentes para ejecutar el proyecto, a pesar de no haber sido la plataforma principal que se usó.

7.3. Software de libre distribución.

El software libre tiene como objetivo fundamental transmitir los conocimientos del programador al resto del mundo de una forma gratuita y fácil de conseguir. Esta es la característica que provoca que la gente confunda frecuentemente el concepto de software libre con el de software gratuito.

Es cierto que existe una relación completamente directa entre el software libre y el software gratuito, pero este primero implica mucho más que simplemente la obtención gratuita del mismo puesto que resulta ser una de las causas más frecuentes de abaratamiento en los costes de producción software.

Una de las razones que permiten el abaratamiento de los coste es que este tipo de software sirve como prototipo simple de un posible proyecto más complejo que únicamente finalizaría su desarrollo ante una respuesta positiva de los usuarios, de este modo se puede predecir el éxito (o fracaso) de un proyecto ante el público, sin hacer peligrar una gran inversión inicial.

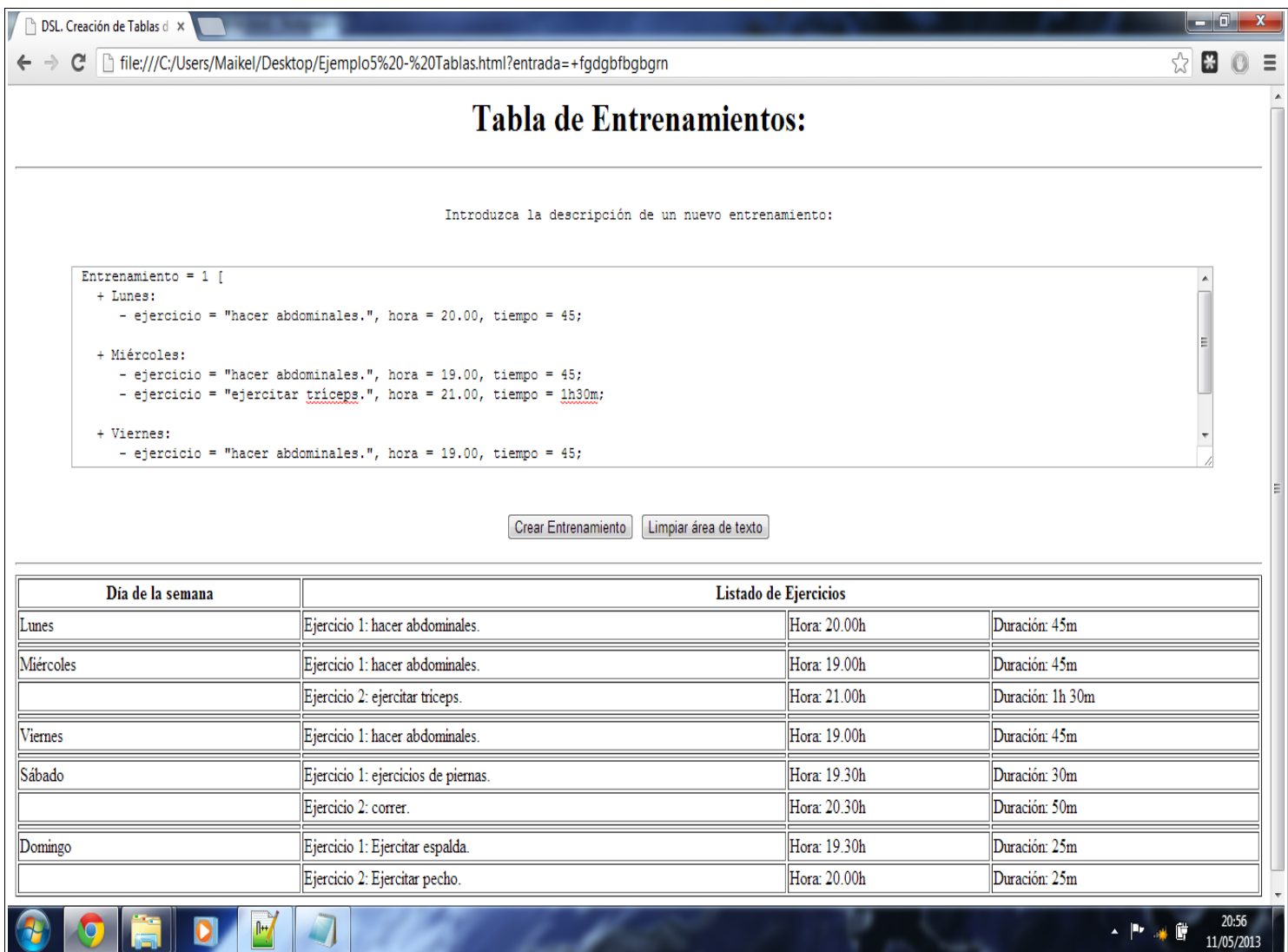
Aunque ésta no fue la motivación de la realización del DSL, cabe añadir que así como se promueve el reciclaje de productos (papel, cartón, latas, vidrio, ...) para retrasar la tala indiscriminada de árboles o la desaparición de parajes naturales como playas en busca de arena para fundir y obtener vidrio, debe también promoverse el desarrollo libre de software para fomentar buenas prácticas que ayudan a evitar “despilfarros” innecesarios en inversiones empresariales de desarrollo software.

7.4. ¿Cómo obtener los códigos fuentes del proyecto?

Los códigos fuentes que permiten la ejecución del proyecto y comenzar a probar la creación de las tablas de entrenamiento, siguiendo la sintaxis del DSL, se encuentran en un repositorio remoto (abierto al público) y que está situado en Git Hub. Se pueden descargar libremente a través de las siguientes URLs:

- GitHub: [git@github.com:maikelp3/DSLTablasEntrenamiento.git](https://github.com/maikelp3/DSLTablasEntrenamiento.git)

8. Primer prototipo.



DSL Creación de Tablas d x

file:///C:/Users/Maikel/Desktop/Ejemplo5%20-%20Tablas.html?entrada=+fgdgbfbbgbrn

Tabla de Entrenamientos:

Introduzca la descripción de un nuevo entrenamiento:

```
Entrenamiento = 1 [  
+ Lunes:  
- ejercicio = "hacer abdominales.", hora = 20.00, tiempo = 45;  
  
+ Miércoles:  
- ejercicio = "hacer abdominales.", hora = 19.00, tiempo = 45;  
- ejercicio = "ejercitar triceps.", hora = 21.00, tiempo = 1h30m;  
  
+ Viernes:  
- ejercicio = "hacer abdominales.", hora = 19.00, tiempo = 45;
```

Crear Entrenamiento Limpiar área de texto

Día de la semana	Listado de Ejercicios		
Lunes	Ejercicio 1: hacer abdominales.	Hora: 20.00h	Duración: 45m
Miércoles	Ejercicio 1: hacer abdominales.	Hora: 19.00h	Duración: 45m
	Ejercicio 2: ejercitar triceps.	Hora: 21.00h	Duración: 1h 30m
Viernes	Ejercicio 1: hacer abdominales.	Hora: 19.00h	Duración: 45m
Sábado	Ejercicio 1: ejercicios de piernas.	Hora: 19.30h	Duración: 30m
	Ejercicio 2: correr.	Hora: 20.30h	Duración: 50m
Domingo	Ejercicio 1: Ejercitar espalda.	Hora: 19.30h	Duración: 25m
	Ejercicio 2: Ejercitar pecho.	Hora: 20.00h	Duración: 25m

20:56
11/05/2013

Proyecto Final: diseño de un DSL.

9. Referencias.