

UNIVERSIDAD DE SANTIAGO DE CHILE

FACULTAD DE INGENIERÍA

DEPARTAMENTO DE INFORMÁTICA



MANUAL DE USUARIO CHATBOT

Programación en Scheme

Nombre Alumno:	Francisco Guajardo
Profesores:	Daniel Gacitúa
Ayudantes:	Javier Vasquez
Fecha de Entrega:	23 – 04 - 2018

Tabla de contenido

INTRODUCCIÓN.....	2
INSTRUCCIONES PREPARACIÓN AMBIENTE TRABAJO	3
FUNCIONALIDADES DEL CHATBOT Y COMO INICIARLO	6
USO DEL CHATBOT.....	8
TEST, CASOS EXITOSOS Y CASOS FALLIDOS.	9

Introducción

El **chatbot** se puede definir como una entidad artificial de conversación. Un programa computacional capaz de mantener una conversación con un ser humano, la que poseerá un contexto y tiempo como delimitador de conocimiento.

Este chatbot esta diseñado para mantener una conversación corta con un usuario que desea comprar una película, juego o manga de la tienda X, indicando su precio y pidiendo forma de pago al usuario.

Este programa funciona en **DrRacket**, que es un entorno de desarrollo integrado programado en Racket, que nos facilitará la tarea de programar en Scheme con las herramientas que ofrece este lenguaje.

Scheme es un lenguaje de programación que funciona en el paradigma funcional y un dialecto de Lisp.

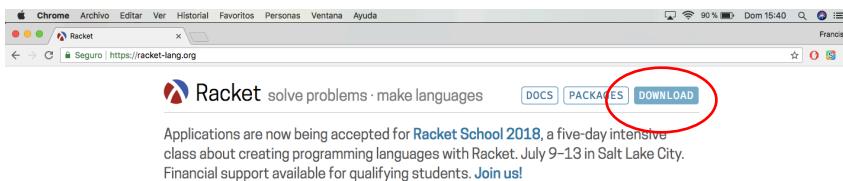
La filosofía del **paradigma funcional** es el uso de funciones para desarrollar algoritmos, este paradigma se basa en el **calculo lambda**. Este calculo es un sistema formal diseñado para la investigación de la definición de función.

Instrucciones preparación ambiente trabajo

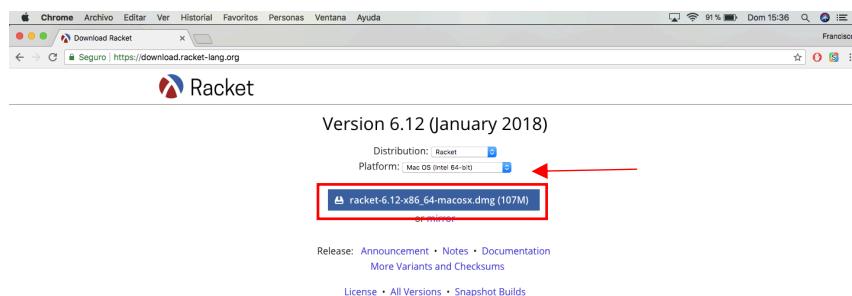
Primero que nada, se debe tener instalado el interprete DrRacket en el PC, este interprete esta disponible tanto en Windows, Mac y Linux.

A continuación, se darán las instrucciones para descargar e instalar DrRacket en macOS.

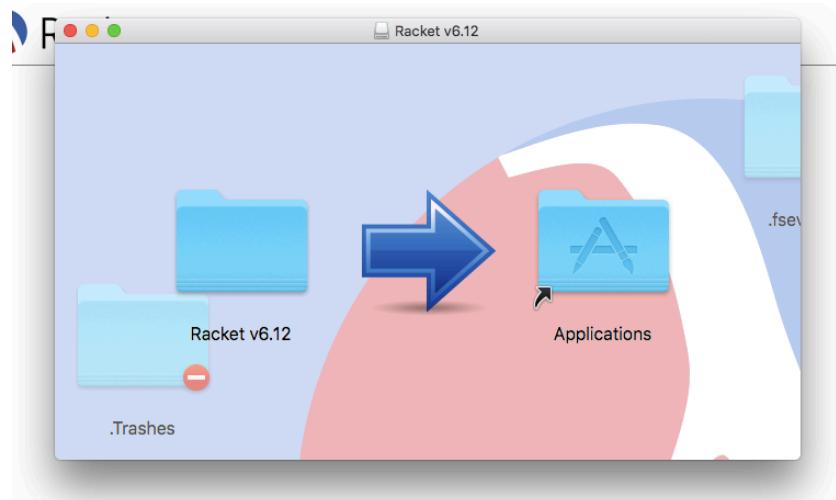
1. Ingresamos a la pagina de Racket <https://racket-lang.org/> y presionamos el botón de Download, como se muestra en la figura:



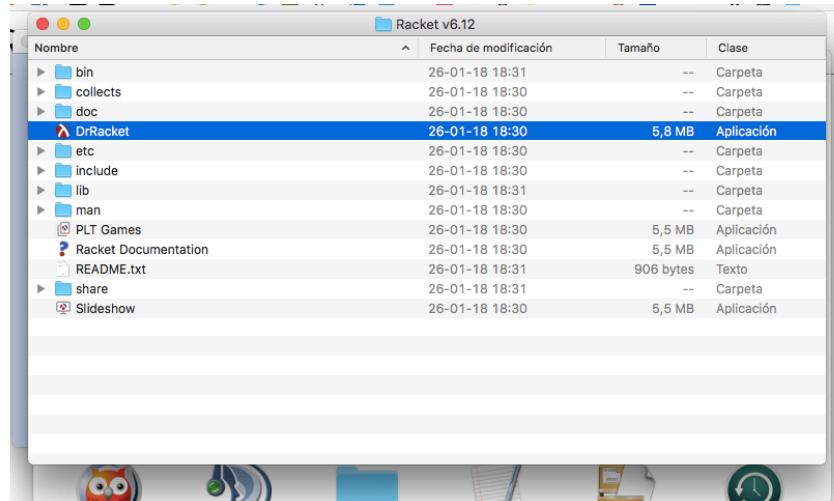
2. Luego, seleccionamos nuestro sistema operativo y presionamos el botón de descarga:



3. Luego, al ya haber descargado el Archivo, abrimos este ultimo, que en este caso se llama: *racket-6.12-x86_64-macosx 15-37-52-427.dmg* y arrastramos la carpeta *Racket v6.12* a la carpeta del sistema *Applications*:

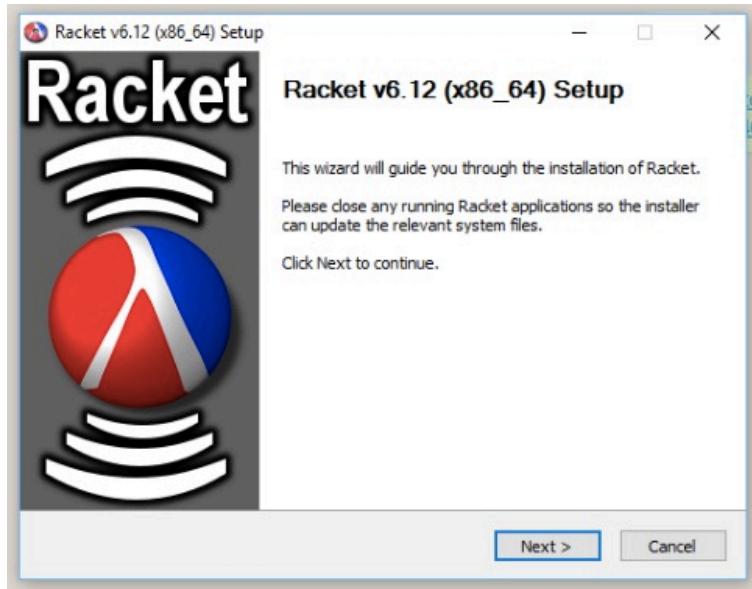


- Al abrir la carpeta que acabamos de arrastrar aparecerán varios archivos, pero el que nos importa es el *DrRacket* y al encontrar este archivo, doble click y ya estaremos en el ambiente de trabajo de Racket:

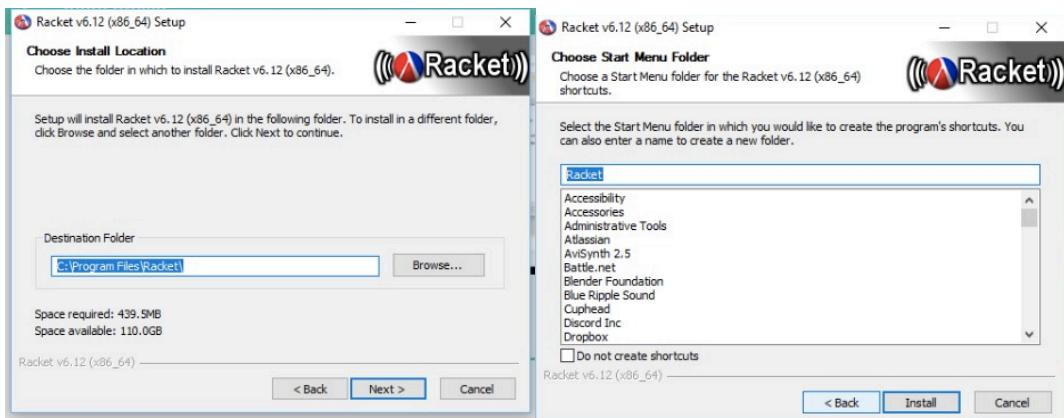


Para sistemas operativos de Windows, los primeros 2 pasos son similares, luego:

- Al abrir el archivo .exe que descargamos, saldrá una ventana similar a la de la siguiente imagen:

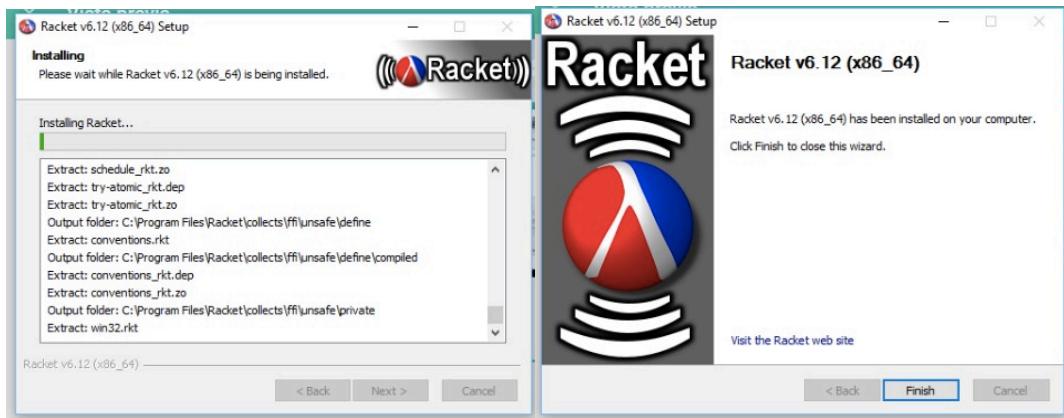


En esta ventana presionamos el botón **Next >** a continuación nos saldrá la opción de donde instalar el DrRacket.



Elegimos el directorio donde instalaremos DrRacket y presionamos en **Next >**. Luego, nos saldrá una nueva ventana y presionamos en **Install**.

Posteriormente, esperamos a que la instalación termine y presionamos en **Finish**

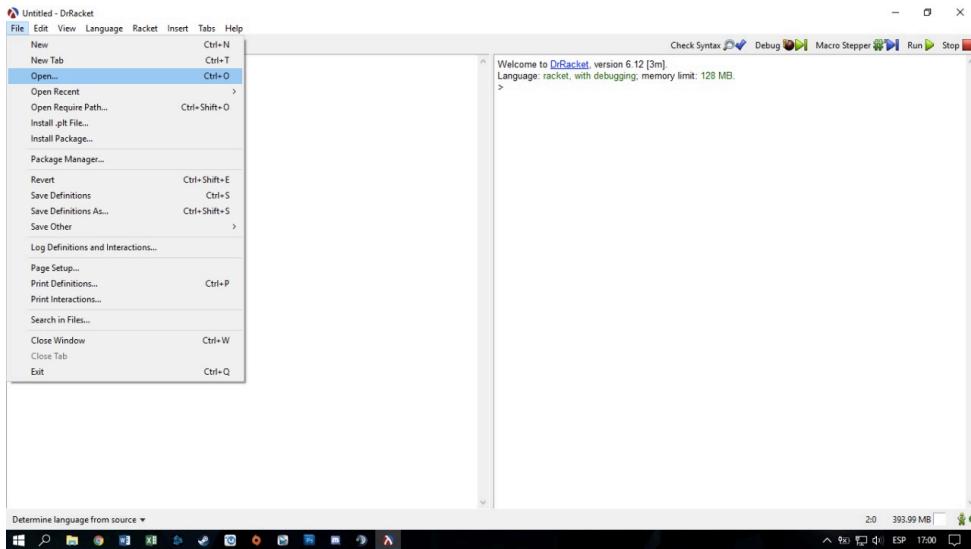


Posterior a eso, buscamos DrRacket en la barra de búsqueda de Windows e iniciamos el programa.

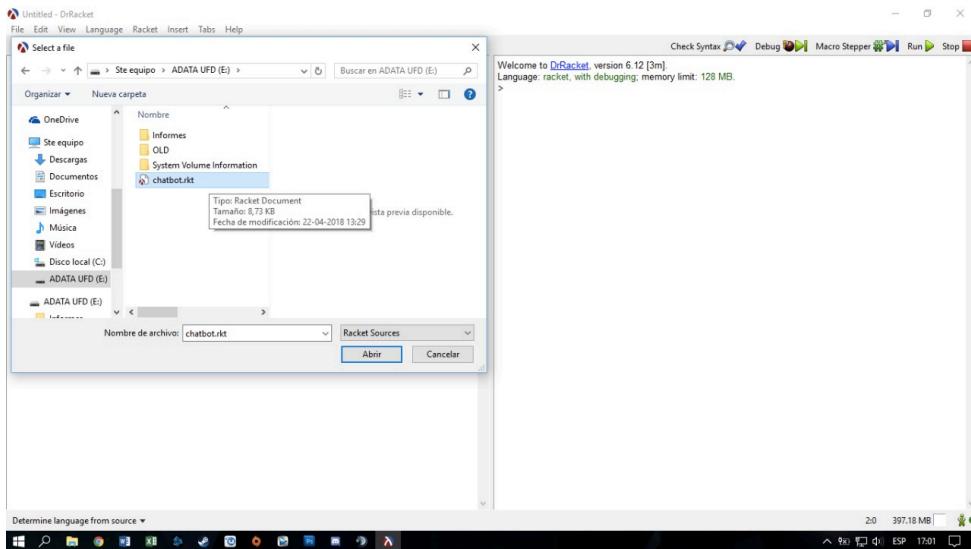
Funcionalidades del Chatbot y como iniciararlo

El chatbot que usted posee tiene una cierta cantidad de funcionalidades y recursos que se deben conocer, su modo de uso se describirá a continuación:

Para iniciar el chatbot, primero debemos tener abierto nuestro interprete DrRacket, como se ilustra en la figura debemos seleccionar la opción File/Archivo -> Open/Abrir.



Luego seleccionar el archivo del chatbot, que en este caso es *chatbot.rkt*



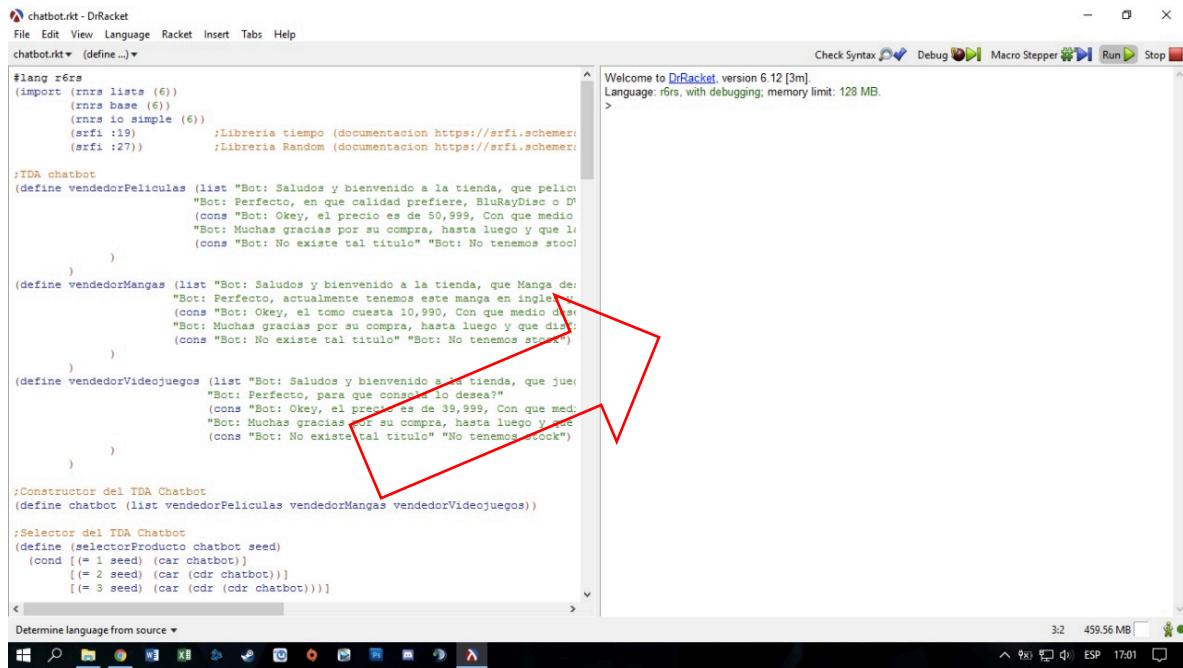
Ya abierto el chatbot podemos proseguir con la explicación de las funciones de este, intentando guiar el camino del usuario para el buen funcionamiento de este programa.

Existe `beginDialog` que es la función que iniciará la conversación con nuestro chatbot, la cual recibirá como parámetros el chatbot en si, el log, o registro histórico de la conversación y una semilla *seed* la cual nos indicara a que vendedor nos estamos refiriendo, siendo números del 1 al 3 los cuales indican si estamos hablando con un vendedor de películas (1), de mangas (2) o de videojuegos (3).

Esta la función `sendMessage` que es la encargada de enviar su mensaje a la maquina para una respuesta del bot, esta función recibe como parámetros los elementos antes mencionados sumándoles su mensaje.

La función `endDialog` nos indicará cuando el dialogo con el chatbot habrá acabado, esta función recibe como parámetros los mencionados en `beginDialog`.

En el interprete, estas funciones se deben ingresar en la ventana inferior o derecha. (esto depende de la configuración del interprete).



```
#lang r6rs
(import (rnrs lists (6))
        (rnrs base (6))
        (rnrs io simple (6))
        (srfi :19)           ;Libreria tiempo (documentacion https://srfi.schemers.org/srfi-19.html)
        (srfi :27))          ;Libreria Random (documentacion https://srfi.schemers.org/srfi-27.html)

;TDA chatbot
(define vendedorPeliculas (list "Bot: Saludos y bienvenido a la tienda, que pelicula te gustaria ver?" "Bot: Perfecto, en que calidad prefieres, BluRay/Disc o Dvd" "Bot: Okey, el precio es de 50,999, Con que medio deseas verla?" "Bot: Muchas gracias por su compra, hasta luego y que disfrutes de tu pelicula" "Bot: No existe tal titulo" "Bot: No tenemos stock"))
(define vendedorMangas (list "Bot: Saludos y bienvenido a la tienda, que Manga deseas leer?" "Bot: Perfecto, actualmente tenemos este manga en ingles" "Bot: Okey, el tomo cuesta 10,990, Con que medio deseas leerlo?" "Bot: Muchas gracias por su compra, hasta luego y que disfrutes de tu manga" "Bot: No existe tal titulo" "Bot: No tenemos stock"))
(define vendedorVideojuegos (list "Bot: Saludos y bienvenido a la tienda, que juego deseas comprar?" "Bot: Perfecto, para que consigues lo deseas?" "Bot: Okey, el precio es de 39,999, Con que medio deseas jugarlo?" "Bot: Muchas gracias por su compra, hasta luego y que disfrutes de tu videojuego" "Bot: No existe tal titulo" "Bot: No tenemos stock"))

;Constructor del TDA Chatbot
(define chatbot (list vendedorPeliculas vendedorMangas vendedorVideojuegos))

;Selector del TDA Chatbot
(define (selectorProducto chatbot seed)
  (cond [(= 1 seed) (car chatbot)]
        [(= 2 seed) (car (cdr chatbot))]
        [(= 3 seed) (car (cdr (cdr chatbot)))]))

Welcome to DrRacket, version 6.12 [3m]
Language: r6rs, with debugging; memory limit: 128 MB.
>
```

Uso del Chatbot

Para comenzar a usar al bot se debe tener en cuenta los siguientes pasos:

- 1. Se debe comenzar una conversación, la forma de comenzar es ingresando la siguiente línea de comandos:

```
(define log1 (beginDialog chatbot log seed))
```

Donde seed se reemplaza por el producto que se desea comprar, indicado anteriormente.

- 2. Si se desea hablar con el chatbot, lo primero que responderá es el nombre del producto que se desea, aquí el usuario debe ingresar lo siguiente:

```
(define log2 (sendMessage "Mensaje" chatbot log1 seed))
```

Donde “Mensaje” corresponde a la respuesta que entregara el usuario y seed no debe de cambiar con respecto a la primera entrada

- 3. Posterior a eso, el chat preguntara algo con respecto a ese producto, el usuario debe de responder escribiendo la función de la siguiente manera:

```
(define log3 (sendMessage "Mensaje" chatbot log2 seed))
```

Teniendo en consideración los parámetros anteriormente descritos.

- 4. Luego de eso, el chat preguntara el medio de pago, por lo cual, teniendo en opciones los estándares de cualquier tienda (Tarjeta, efectivo, cheque, etc) el usuario debe de hacer ingreso de su respuesta de la forma:

```
(define log4 (sendMessage "Mensaje" chatbot log3 seed))
```

Usando las variables de la forma anteriormente descrita.

- 5. Para finalizar la conversación, ya en este punto habrá terminado, se debe de incluir la siguiente línea:

```
(define log5 (endDialog chatbot log4 1))
```

Respetando siempre el ingreso y formato de las variables ya descritas.

Test, casos exitosos y casos fallidos.

Existe una función `test` que prueba el chatbot con una serie de mensajes ya predefinidos en el código, su funcionamiento es el siguiente:

```
(test user chatbot log seed)
```

El cual recibe un user que puede ser user1, user2, user3 y la seed perteneciente a cada user, (ejemplo user1 con seed 1). Sirve para comprobar que el chatbot no fallará o por si algún error se comete al ingresar el usuario sus datos no sea error de código.

-Caso exitoso, si se sigue todas las reglas antes dichas el programa debería de funcionar correctamente:

```
> (define log1 (beginDialog chatbot log 1))
> (define log2 (sendMessage "The Last Jedi" chatbot log1 1))
> (define log3 (sendMessage "BlueRay" chatbot log2 1))
> (define log4 (sendMessage "Efectivo" chatbot log3 1))
> (define log5 (endDialog chatbot log4 1))
```

Esas instrucciones deberían de generar un registro de conversación del tipo:

- Bot: Saludos y bienvenido a la tienda, ¿que película desea comprar?
- User: The Last Jedi
- Bot: Perfecto, en que calidad prefiere, ¿BluRayDisc o DVD?
- User: BlueRay
- Bot: Okey, el precio es de 50,999, ¿Con que medio desea pagar?
- User: Efectivo
- Bot: Muchas gracias por su compra, hasta luego y que la disfrute

Y en el registro queda el `beginDialog` y `endDialog`.

-Caso de error, existen errores de tipo, ingreso mal variable, etc.

```
> (define log1 (beginDialog chatbot log 5))
Error al ingresar semilla, asegurese de ingresar un numero entero entre 1 y 4 incluidos
```

En este caso solo se debe de seguir las instrucciones mostradas en pantalla.

-Caso error, cuando se ingresa mal el nombre de una variable. En estos casos solo basta con reescribir la definición revisando detenidamente los nombres de las variables ingresadas.

Ejemplos de como se mostraría un error.

```
> (define log1 (beginDialog chaybot log 3))
✖✖ chaybot: undefined;
cannot reference an identifier before its definition
> (define log1 (begindialog chatbot log 3))
✖✖ begindialog: undefined;
cannot reference an identifier before its definition
```