Familiarização com o ambiente de desenvolvimento Racket

Prof. Diogo S. Martins

santana.martins@ufabc.edu.br

MCTA016 - Paradigmas de Programação (Prática)

05 de junho de 2018

UFABC

CMCC
Centro de Matemática, Computação e Cognição

Crédito de parte das imagens, a menos se especificado: Wikipedia

Objetivos

- Compreender o ambiente de desenvolvimento típico da linguagem Racket
- Aprender a configurar as principais ferramentas do ambiente
- Testar o ambiente que foi configurado

A família Lisp

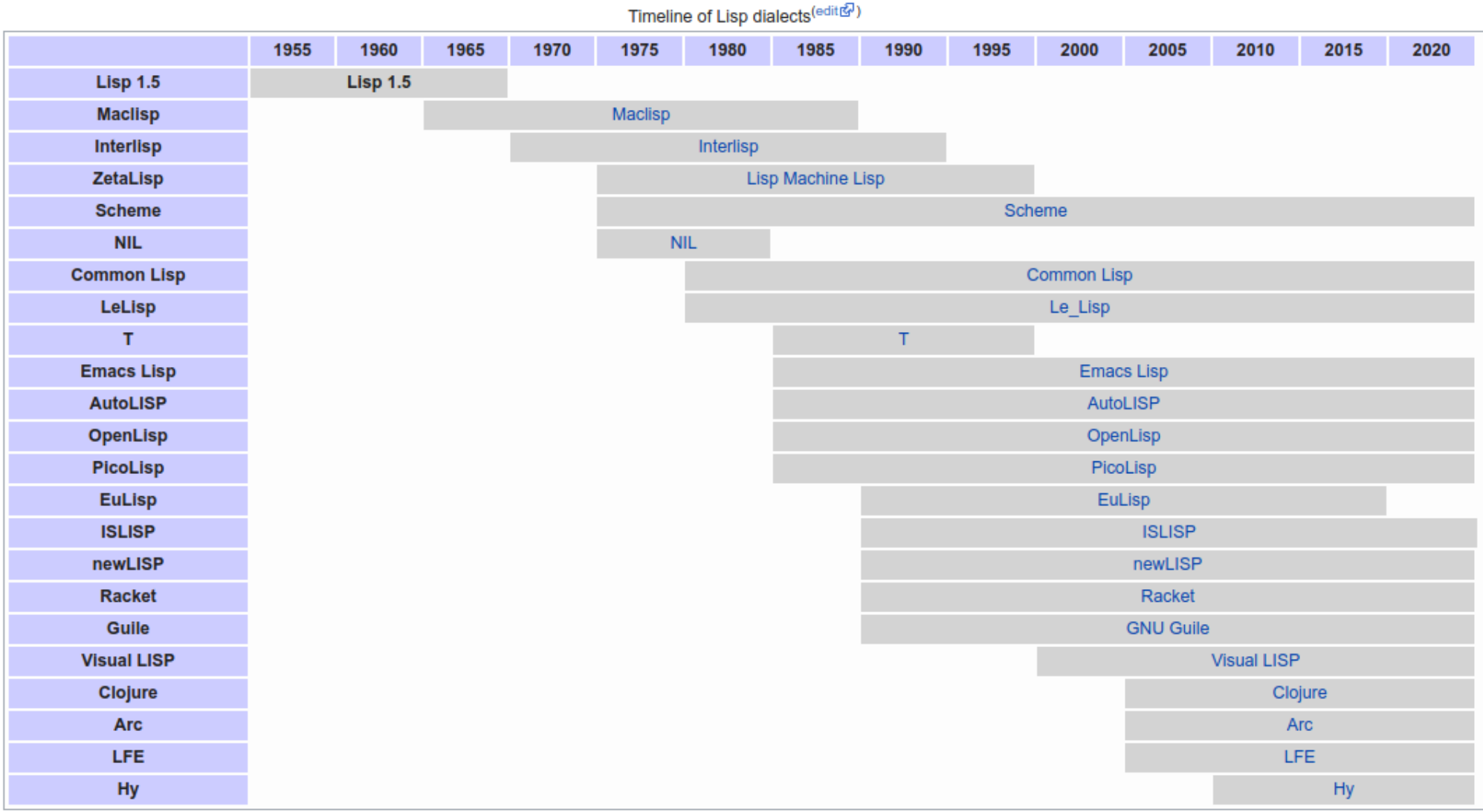
- **LIS**t **P**rocessor
- Proposta em 1958 (John McCarthy, MIT)
  - *"Recursive Functions of Symbolic Expressions and Their Computation by Machine, Part I"*. URL: `https://dl.acm.org/citation.cfm?id=367199`
- Inspiração no Cálculo-λ (Alonzo Church)
- Notação de listas (S-expressions) para representar código e dados

```
1 ((lambda (arg) (+ arg 2)) 6)
```

- Manipula código como estruturas de dados (propicia extensões de sintaxe, DSLs, etc.)
- Linguagem mínima, com extensões para vários estilos de programação (imperativa, funcional, OO, lógica, etc.)

Dialetos Lisp

Timeline



https://en.wikipedia.org/wiki/Lisp_(programming_language)#Timeline

Scheme

- Criada em 1970, no MIT AI Lab
- Criadores: Guy Steele e Gerald Sussman
- Motivação: implementação do *Actor Model* (Carl Hewitt) para sistemas concorrentes
- Popularizada pelos relatórios chamados *Lambda Papers*
- Padronizada oficialmente via IEEE 1178-1990 (R1995)
- Padronizada pela comunidede via R^nRS
  - Mais recente: R^6RS
  - Mais difundida nas implementações: R^5RS
  - Em construção: R^7RS

Racket

- Proposta em 1995 (Mathias Felleisen)
- Versões iniciais: PLT Scheme, utilizada com propósitos didáticos
- Versão 5.0 (2010) culminou na mudança de nome (Racket)
- Altamente extensível via macros (i.e extensão de sintaxe, DSLs, etc.)
- Dezenas de linguagens-base disponíveis

Racket

Principais linguagens-base

#lang racket

```
(require 2htdp/image) ; draw a picture
(let sierpinski ([n 8])
  (cond
    [(zero? n) (triangle 2 'solid 'red)]
    [else (define t (sierpinski (- n 1)))
```

#lang scribble/base

```
@; Generate a PDF or HTML document
@title{Bottles: @italic{Abridged}}
@(apply
  itemlist
  (for/list ([n (in-range 100 0 -1)])
```

#lang typed/racket

```
;; Using higher-order occurrence typing
(define-type SrN (U String Number))
(: tog ((Listof SrN) -> String))
(define (tog l)
  (apply string-append (filter string? l)))
```

#lang datalog

```
ancestor(A, B) :- parent(A, B).
ancestor(A, B) :-
  parent(A, C), ancestor(C, B).
parent(john, douglas).
parent(bob, john).
```

#lang racket/gui

```
(define f (new frame% [label "Guess"]))
(define n (random 5)) (send f show #t)
(define ((check i) btn evt)
  (message-box "." (if (= i n) "Yes" "No")))
(for ([i (in-range 5)])
```

#lang web-server/insta

```
;; A "hello world" web server
(define (start request)
  (response/xexpr
   '(html
     (head (title "Racket"))
```

Ambiente de desenvolvimento Scheme

Componentes

- Interpretador e/ou compilador
  - Chez Scheme, Gambit, Guile, MIT Scheme, Racket, Chicken Scheme, etc.
- Editor de texto (com facilidades para Scheme)
  - Emacs, Vim, Sublime, Atom, Visual Studio Code, etc.
- IDEs
  - DrRacket (multi-dialetos), Emacs+Quack, Emacs+ChickenSlime, etc.

Nota sobre a plataforma de desenvolvimento

A plataforma de desenvolvimento oficial da disciplina é estruturada sobre o sistema operacional **Ubuntu Linux**, na versão disponível nos laboratórios. Todas as orientações e exemplos da aula serão baseados nos recursos dessa plataforma.

Você pode utilizar outra plataforma de sua preferência, por exemplo estruturada no Windows, porém as adaptações necessárias ficam por sua conta.

O mesmo vale para editores de código, a escolha de qual usar fica a seu critério, contanto que as convenções de indentação sejam respeitadas (as quais veremos na próxima aula).

Instalação do ambiente Racket

Instalação do ambiente Racket

- Download em: `https://download.racket-lang.org/`
- Instalar via PPA:
  `https://launchpad.net/~plt/+archive/ubuntu/racket`

```
sudo add-apt-repository ppa:plt/racket
sudo apt-get update
sudo apt install racket
```

Interpretador REPL: racket

- Após instalação testar o interpretador:

```
$ racket
Welcome to Racket v6.12.
>
```

- Alguns expressões de teste:

```
> "Hello World"
"Hello World"
> (define (hello who) (string-append "Hello " who "!"))
> (hello "World")
"Hello World!"
> (+ 1 2)
3
> (define (sum a b) (+ a b))
> (sum 1 2)
3
> (car (cdr '(1 2 3)))
2
> ,quit
```

- Documentação do interpretador:
  https://docs.racket-lang.org/guide/racket.html

Interpretador: `racket`

- Podemos executar também como *script*
- Crie um arquivo com o código (`hello.rkt`):

```
1 #lang racket
2
3 (define (hello who) (string-append "Hello " who "!"))
4 (hello "World")
```

- Execute parametrizando o interpretador:

```
$ racket test.rkt
"Hello World!"
```

- O interpretador Racket é uma máquina virtual que executa bytecode
- A chamada acima:
  1. Compila para bytecode
  2. Executa o bytecode na máquina virtual

`raco`: Racket command line tools

- `raco` permite executar várias tarefas, entre as principais:
  - `rako make`: compila para bytecode
  - `raco exe`: gera executáveis
  - `raco distribute`: gera executáveis estáticos
  - `raco pkg`: gerenciamento de pacotes
  - `raco setup`: gerenciamento de instalações
  - etc.

raco: Racket command line tools

```
$ raco help
Usage: raco <command> <option> ... <arg> ...

Frequently used commands:
  docs                  search and view documentation
  make                  compile source to bytecode
  setup                 install and build libraries and documentation
  pkg                   manage packages
  planet                manage Planet package installations
  exe                   create executable
  test                  run tests associated with files/directories

All available commands:
  check-requires        check for useless requires
  contract-profile      profile overhead from contracts
  ctool                 compile and link C-based extensions
  decompile             decompile bytecode
  demodularize          produce a whole program from a single module
  dependencies-graph    opens a GUI window showing transitive module dependencies (aka `Module
  distribute            prepare executable(s) in a directory for distribution
  docs                  search and view documentation
  exe                   create executable
  expand                macro-expand source
  link                  manage library-collection directories
  macro-profiler        profile macro expansion (code size)
  macro-stepper         explore expansion steps
  make                  compile source to bytecode
  pack                  pack files/collections into a .plt archive
  pkg                   manage packages
  planet                manage Planet package installations
  profile               profile execution time
  read                  read and pretty-print source
  scribble              render a Scribble document
  setup                 install and build libraries and documentation
  show-dependencies     show module dependencies
```

`raco make`: compilação para bytecode

- `raco make`: compila para bytecode
- Propicia execução mais eficiente do que o interpretador puro

```
$ raco make hello.rkt
$ ls compiled/
hello_rkt.dep  hello_rkt.zo
$ racket hello.rkt
"Hello World!"
```

- O interpretador automaticamente usa o bytecode da pasta `compiled` (não compila novamente no início da execução)

`raco exe`: geração de executáveis

- `raco make`: gera um executável para a plataforma
- Na prática, encapsula o interpretador `racket` com execução automática do programa e suas dependências

```
$ raco exe hello.rkt
$ ls -lh | awk '{print $5 "\t" $9}'
4.0K  compiled
6.8M  hello
83    hello.rkt
```

- O tamanho do executável deve-se à inclusão do interpretador
- Ainda necessita do ambiente Racket instalado para funcionar (i.e. não inclui as dependências)

`raco distribute`: geração de executáveis estáticos

- `raco distribute`: gera uma distribuição do programa, específica para a plataforma atual
- Distribui o programa e suas dependências na estrutura de diretórios convencional da plataforma

```
$ raco distribute hello-dist hello
$ find hello-dist/
hello-dist/
hello-dist/lib
hello-dist/lib/plt
hello-dist/lib/plt/racket3m-6.12
hello-dist/lib/plt/hello
hello-dist/lib/plt/hello/exts
hello-dist/lib/plt/hello/collects
hello-dist/bin
hello-dist/bin/hello
```

- Com essa estrutura podemos fazer um script que instala a distribuição

`raco pkg`: instalação de pacotes

- `raco pkg`: instala pacotes, que podem ser bibliotecas, aplicações, linguagens, interpretadores, etc., desenvolvidos pela comunidade Racket
- Listar os pacotes instalados:

```
$ raco pkg show
Installation-wide:
 Package            Checksum             Source
 main-distribution  fd230051d9990fa5...  catalog...tribution
 racket-lib         810b69bd48b5b211...  catalog racket-lib
 [192 auto-installed packages not shown]
User-specific for installation "6.12":
 [none]
```

- Para uma listagem mais completa:

```
$ raco pkg show --all
Installation-wide:
 Package[*=auto]              Checksum          Source
 2d*                          60d96aad8b1d5...  catalog 2d
 2d-doc*                      e16514a841aa5...  catalog 2d-doc
 2d-lib*                      5e607f0a11ff0...  catalog 2d-lib
 algol60*                     18268cdcb837f...  catalog algol60
 at-exp-lib*                  39aa817ddf1fd...  catalog...xp-lib
 base*                        842e2b328663a...  catalog base
 cext-lib*                    55d25f6eef980...  catalog cext-lib
 class-iop-lib*               3887471740eca...  catalog...op-lib
 compatibility*               1e0f09985626c...  catalog...bility
 compatibility-doc*           10d4c9d0271d8...  catalog...ty-doc
 compatibility-lib*           6e7c6038296d8...  catalog...ty-lib
 compiler*                    0e4be9aa09394...  catalog compiler
```

Repositório de pacotes

- Repositório oficial: `https://pkgs.racket-lang.org/`

970 packages (see all, including packages tagged as "deprecated", "main-distribution", or "main-test")

593 todos. Click here to see them.

Package	Description	Build
2htdp-typed lexi.lambda@gmail.com	A partial version of 2htdp for Typed Racket **Docs:** 2htdp-typed **Tags:** 2htdp big-bang gui typed-racket	succeeds
3d-model code_man@cybnet.ch	3d-model vertex/tri representaion library **Docs:** 3d-model **Tags:** 3d model	succeeds; has dependency problems
3s jay.mccarthy@gmail.com	positional sound and mixing for lux and other programs **Docs:** 3s **Tags:** audio games k s	fails
_ leif@leifandersen.net	This package needs a description **Docs:** _	succeeds

Instalação de pacotes

- Se o pacote estiver no repo oficial, podemos utilizar o `raco` para baixar e instalar
- Por exemplo, vamos instalar uma biblioteca para E/S binária:

```
$ raco pkg install binaryio
Resolving "binaryio" via https://download.racket-lang.org/releases/6.12/catalog/
Resolving "binaryio" via https://pkgs.racket-lang.org
Downloading repository git://github.com/rmculpepper/binaryio?path=binaryio
The following uninstalled packages are listed as dependencies of binaryio:
   binaryio-lib
Would you like to install these dependencies? [Y/n/a/c/?]
Resolving "binaryio-lib" via https://download.racket-lang.org/releases/6.12/catalog/
Resolving "binaryio-lib" via https://pkgs.racket-lang.org
Using cached1528220774152822077447 3 for git://github.com/rmculpepper/binaryio?path=bina
The following uninstalled packages were listed as dependencies
and they were installed:
 dependencies of binaryio:
   binaryio-lib
raco setup: version: 6.12
(...)
raco setup: 2 rendering: <pkgs>/racket-index/scribblings/main/user/local-redirect.scrbl
raco setup: 1 rendering: <pkgs>/racket-index/scribblings/main/user/release.scrbl
raco setup: 0 rendering: <pkgs>/racket-index/scribblings/main/user/search.scrbl
raco setup: 1 rendering: <pkgs>/racket-index/scribblings/main/user/start.scrbl
raco setup: --- installing collections ---
raco setup: --- post-installing collections ---
```

- Instalou no userspace; para instalação global, usar `sudo`

Testando o pacote instalado

- Ler um stream de caracteres byte a byte:

```
$ racket
Welcome to Racket v6.12.
> (require binaryio/bytes)
> (define in (open-input-bytes #"abcde"))
> (read-bytes* 4 in)
#"abcd"
> (read-bytes* 2 in)
; read-bytes*: unexpected end of input
;   tried to read: 2 bytes
;   available: 1 bytes
;   received: #"e"
; [,bt for context]
```

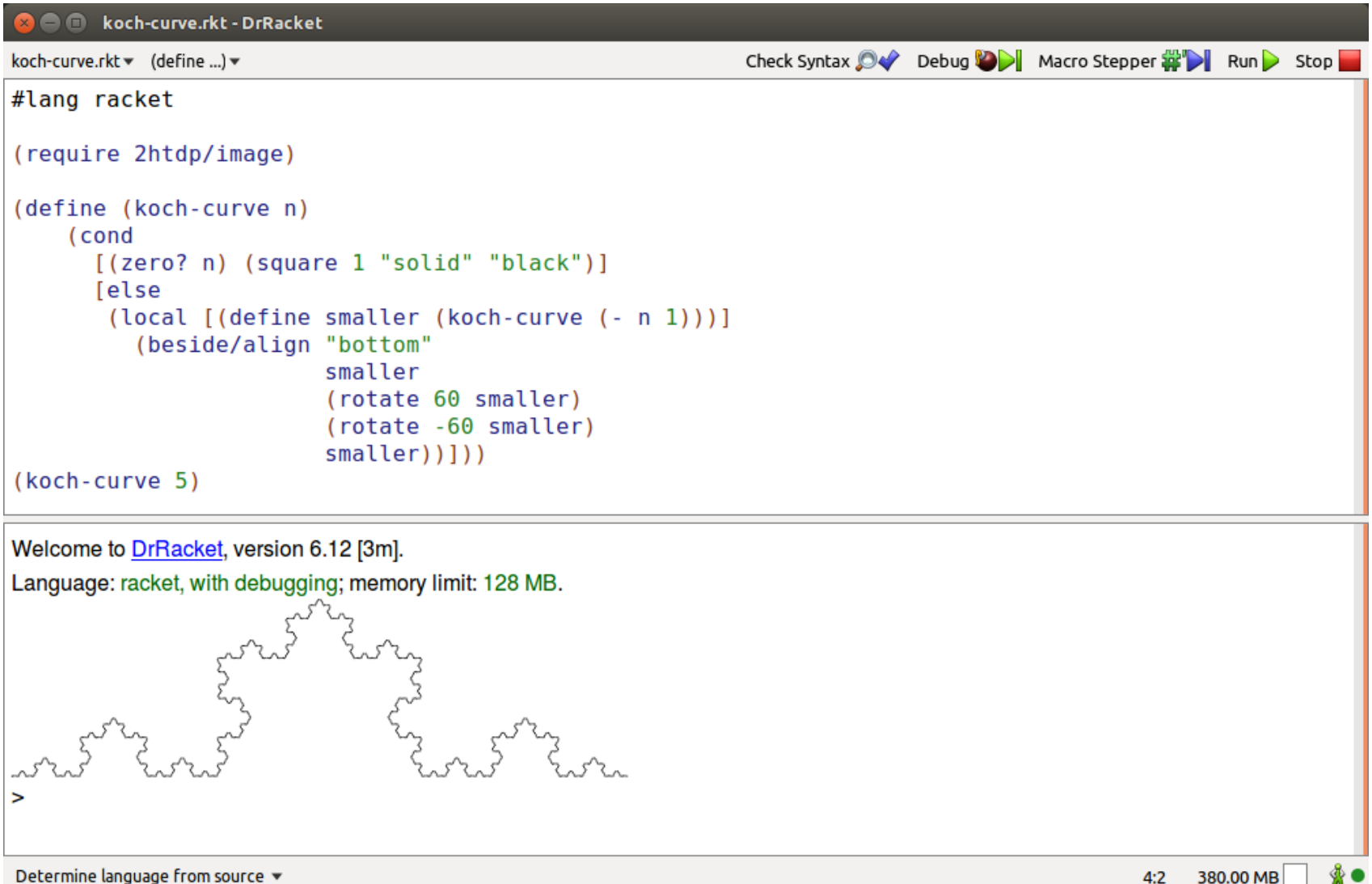
Editores de código

Editores para Scheme e Racket

- Objetivo: editores com indentação automática e que tenham *syntax highlighting*
- Escolha do editor é livre nessa disciplina
  - Ressalva: as convenções de indentação devem ser seguidas! (veremos na próxima aula de lab)
- Alternativas:
  - Emacs
  - Emacs+Quack
  - Dr.Racket
  - Vim
  - Sublime
  - Atom
  - etc.

DrRacket

- IDE oficial, incluída na instalação
- Iniciar via menu de aplicações ou Ubuntu ou linha de comando (`drracket`)

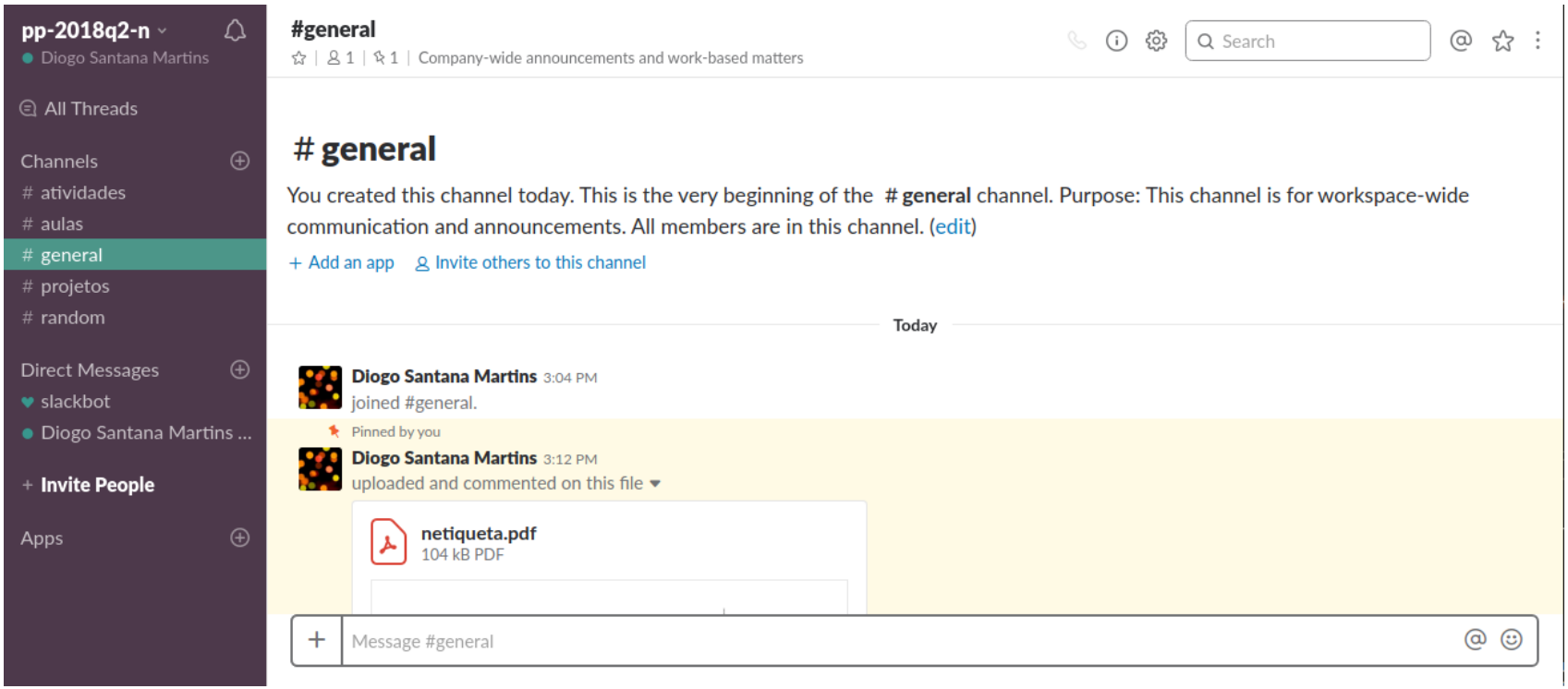


Emacs, Vim, Sublime, VSCode

- Emacs: `https://docs.racket-lang.org/guide/Emacs.html`
- Vim: `https://docs.racket-lang.org/guide/Vim.html`
- Sublime: `https://packagecontrol.io/packages/Racket`
- Visual Studio Code: `https://marketplace.visualstudio.com/items?itemName=karyfoundation.racket`

Worksite Slack

- Não esqueça de se increver no nosso worksite do Slack
- Link de convite: `https://bit.ly/2xLp7kp`
- Vamos usar para comunicação interna na disciplina



Materiais complementares

- Dialects of Racket and Scheme. URL: `https://docs.racket-lang.org/guide/dialects.html`
- Command-Line Tools and Your Editor of Choice. URL: `https://docs.racket-lang.org/guide/other-editors.html`
- `raco`: Racket Command-Line Tools. URL: `https://docs.racket-lang.org/raco/index.html`
- Package Management in Racket. URL: `https://docs.racket-lang.org/pkg/index.html`
- Slack for new members: `https://get.slack.help/hc/en-us/articles/218080037-Getting-started-for-new-members`

Familiarização com o ambiente de desenvolvimento Racket

Prof. Diogo S. Martins

santana.martins@ufabc.edu.br

MCTA016 - Paradigmas de Programação (Prática)

05 de junho de 2018

UFABC

CMCC
Centro de Matemática, Computação e Cognição

Crédito de parte das imagens, a menos se especificado: Wikipedia