Rascal Cheat Sheet

by rubene via cheatography.com/19109/cs/2146/

Default datatypes

String	str
Boolean	bool
Array	list[**T**]
Location	loc
Map / set	map[**T**, **T**]
Integer	int
Set (unique array)	set[**T**]

T = type. For example: list[str] for a list of strings

Imports

String	import String
Boolean	import Boolean
List	import List
Location	import Location
Map	import Map
Integer	import util::Math
Set	import Set
M3 AST	import analysis::m3::AST
IO	import IO

Read java project

```
createAstsFromDirectory(loc project, true);
```

This method is located in the M3 AST library
return: set[Declaration]

Debugging

Print to console (one line)	println(value)
Print to console (formatted)	iprintln(value)

This methods are located in the IO library

Location

File	|file:///home/rascal/rascal.rsc|
Project	|project://android-project/|
URL	|http://www.google.nl|
Folder	|file:///home/rascal/|

Data manipulation 1

String - append	str : "str" + "str"
String - compare	bool : "str == "str"
String - interpolation	"value: <str>"
List - assignment	list[int] : [3,2,1];
List - append	list[int] : [3,2] + 1;
Set - assignment	set[int]: {3,2,1}
Set - append	set[int]: {3,2} + 1;

All values are immutable

Data manipulation 2

```
// data Game = game(list[Player] players);
// Game game = getGame();
// Access players:
game.p layers
// Overwrite players:
game.p layers = []
// Add player:
game.p layers = game.p layers + player
```

If else

```
if("str" == "str") {
        ipr int ln( " tru e");
} else {
        ipr int ln( " fal se");
}
```

For loop (array)

```
list[int] numbers = [1, 2, 3, 4, 5];
for (number <- numbers) {
        ipr int ln( num ber);
}
```

For loop (map)

```
map[str, int] numbers = ["one" : 1, "two" : 2,
" thr ee" : 3, " fou r" : 4, " - fiv e" : 5];
for (numbe rText <- numbers) {
       int number Value = number s[n umb erT ext];
        ipr int  ln ( n um  - ber Value);
}
```

File manipulation

Read file	readFile(location)
Write file	writeFile(location, value)

Method definition

```
public str getString() {
        return " Str ing ";
}
```

The method test() is reserved

Visit pattern - Matches

wildcard	_
Value check	"exact value"
Assignment	method
Block assignment	block:

Visit pattern - Example 1

```
visit (ast) {
       case \impor t(i mpo rtN - ame): {
                ipr int ln( - imp ort Name);
       }
}
```

By **rubene**
cheatography.com/rubene/

Published 28th May, 2014.
Last updated 11th May, 2016.

Visit pattern - Example 2

```
visit (ast) {
       case \impor  t(_): {
                ipr int  ln -
( "  Found an import  ");
       }
}
```

Visit pattern - Example 3

```
visit (ast) {
       case \impor  t( " I m -
po  rte  dC l a ss  "): {
                ipr int  ln -
( "  found Import  ed C l as  -
s");
       }
}
```

Visit pattern - Example 4

```
visit (ast) {
       case \field (si mpl eTy -
pe( sim ple Nam e(n ame)), _): {
                ipr int  ln -
( "  found a field: <na me> ");
       }
}
```

Syntax keywords

layout	The layout of the syntax. Is the syntax separated by spaces or new lines.
lexical	Definition for a block of the input, which is separated by the layout.
start syntax	The global syntax definition of the input
syntax	The global syntax can be separated into smaller pieces, which can be defined with a syntax.

By **rubene**
cheatography.com/rubene/

Published 28th May, 2014.
Last updated 11th May, 2016.