# RASCAL Cheat Sheet

www.meta-environment.org/Rascal

## Module

```
module N
import N;
...
Declarations
...
```

*Declarations may be* private *(not visible outside M) or* public.

## Declarations

| | |
|---|---|
| import *N* | Import module *N* |
| data *N* = *P* \| ... | Data type *N* |
| *T N*(*T V*, ...) *S*<br>*T N*(*T V*, ...) *S*<br>throws *N*, *N* | Function *N* |
| *T V* = *E* | Variable *V* |
| alias *N* = *T* | Alias *N* |
| anno *T T* @ *N* | Annotation *N* |
| rule *N P* => *P*<br>rule *N P* : *S* | Rewrite rule *N* |

## Control Statements

| |
|---|
| if(*E*) *S* |
| if(*E*) *S* else *S* |
| while(*E*) do *S* |
| do *S* while(*E*) |
| for(*E*,*E*,...) *S* |
| switch(*E*) {<br>case *P* => *P*<br>case *P* : *S*<br>...<br>default: *S*<br>} |
| try *S*<br>catch *P* => *P*<br>catch *P* : *S*<br>finally: *S* |
| throw *E* |
| fail |
| return<br>return *E* |
| solve(*V*, *V*, ...) *S* |

## Other statements

| |
|---|
| { *S*; *S*; ... } |
| assert *E*<br>assert *E* : *E* |
| test *E*<br>test *E* : *E* |
| append *E* |
| insert *E* |

## Types and Values

| | |
|---|---|
| bool | true, false |
| int | 1, 0, -1, 123456789 |
| real | 2.30E-14 |
| str | "rascal" |
| loc | \|file:///etc/passwd\| |
| tuple[*T*, ...]<br>tuple[*T V*, ...] | <"monday", 1> |
| list[*T*] | [1,2,3,2,1] |
| set[*T*] | {1,2,3} |
| map[*T*,*T*] | ("mo":1, "tue": 2) |
| rel[*T*, ...]<br>rel[*T V*, ...] | {<2001,5>, <2002, 6>} |
| node | f(), g("abc",[2,3,4]) |
| void, value | |

## Assignment

| |
|---|
| *A* = *E* |
| *A op*= *E* |

## Assignables

| |
|---|
| *V* |
| *A*[*E*] |
| *A*.*N* |
| <*A*,*A*,...> |
| *A* ? *E* |
| *A* @ *N* |
| *N*(*A*,*A*,...) |

## Expressions

| | |
|---|---|
| *E op E* | Binary expression |
| *op E* | Prefix expression |
| *E op* | Postfix expression |
| *E* . *N* | Field selection |
| *E* [*N* = *E*] | Field assignment |
| *E* < *N*, ... > | Field projection |
| *E*[*E*, ...] | Subscription |
| *E*[@*N* = *E*] | Annotation replacement |
| *N*(*E*,*E*,...) | Function call |
| *E* ? *E* : *E* | Conditional expression |
| [*E* .. *E*]<br>[*E*, *E*, .. *E*] | Range |
| [*E*, ... \| *E*, *E*, ...] | List comprehension |
| {*E*, ... \| *E*, *E*, ...} | Set comprehension |
| (*E* : *E* \| *E*,*E*, ...) | Map comprehension |
| visit(*E*) {<br>case *P* => *P*<br>case *P* : *S*<br>...<br>} | Visit (traversal) |
| if, while, do, for | |

## Infix Operators

| | |
|---|---|
| + | Addition, union, concatenation |
| - | Subtraction, difference |
| * | Multiplication, product |
| / | Division |
| % | Modulo |
| & | Intersection |
| join | Join |
| o | Compose |
| && | And |
| \|\| | Or |
| ==, != | (In)equality |
| <, <=, >, >= | Comparison |
| ==>, <==> | Implies, equivalence |
| in, notin | (Not) element of |
| :=, !:= | Match, no match |
| <- | Enumerator |
| @ | Annotation |

## Postfix Operators

| | |
|---|---|
| + | Transitive closure |
| * | Reflexive transitive closure |
| ? | Is defined |

## Prefix Operators

| | |
|---|---|
| - | Negation |
| ! | Not |

| Abstract Patterns | |
|---|---|
| *T V* | Variable declaration |
| *V** | Multi-variable (list or set) declaration |
| *V* | Variable use |
| [*P, P, ...*] | List |
| {*P, P, ...*} | Set |
| <*P, P, ...*> | Tuple |
| *N*(*P, P, ...*) | Node |
| */ P* | Descendant |
| *V* : *P* | Labeled |
| *T V* : *P* | Typed, Labelled |
| [*T*] *P* | Type constraint |

| Concrete Patterns | |
|---|---|
| ` *Tok Tok ...*` | Quoted |
| *T* ` *Tok Tok ...*` | Typed & Quoted |
| *Tok Tok ...* | Unquoted |
| <*T V*> | Typed variable |

| Regular Expression Patterns | |
|---|---|
| / ... / | Regular Expression |
| <*V* : *R*> | Named RegExp |
| *Tok Tok ...* | Unquoted |
| <*T V*> | Typed variable |

| Legend | |
|---|---|
| *A* | Assignable |
| *E* | Expression |
| *op* | Operator |
| *S* | Statement |
| *V* | Variable |
| *T* | Type |
| *P* | Pattern |
| *N* | Name |
| *R* | Regular Expression |
| *Tok* | Lexical token |

| Standard Library | |
|---|---|
| ATermIO | Read/write values as ATerms |
| Benchmark | Benchmarking tools |
| Boolean | Boolean functions |
| Chart | Chart drawing |
| Exception | Exceptions a program can catch |
| Graph | Graph manipulation |
| Integer | Integer functions |
| IO | Input/output |
| JDT | Java fact extraction functions |
| LabeledGraph | Labeled graph manipualtion |
| List | List functions |
| Location | Location functions |

| Standard Library | |
|---|---|
| Map | Map functions |
| Node | Node functions |
| PriorityQueue | Functions on prio queues |
| Real | Real functions |
| Relation | Functions on relations |
| Resource | Retrieve Eclipse resources |
| RSF | Read RSF files |
| Set | Functions on sets |
| String | Functions on strings |
| Tree | Functions on parse trees |
| Tuples | Functions on tuples |
| ValueIO | Read/write values as text/ binary |
| View | Grap/tree/text display of values |