TABLE OF CONTENTS

1. /domotica [Modules]

NAME

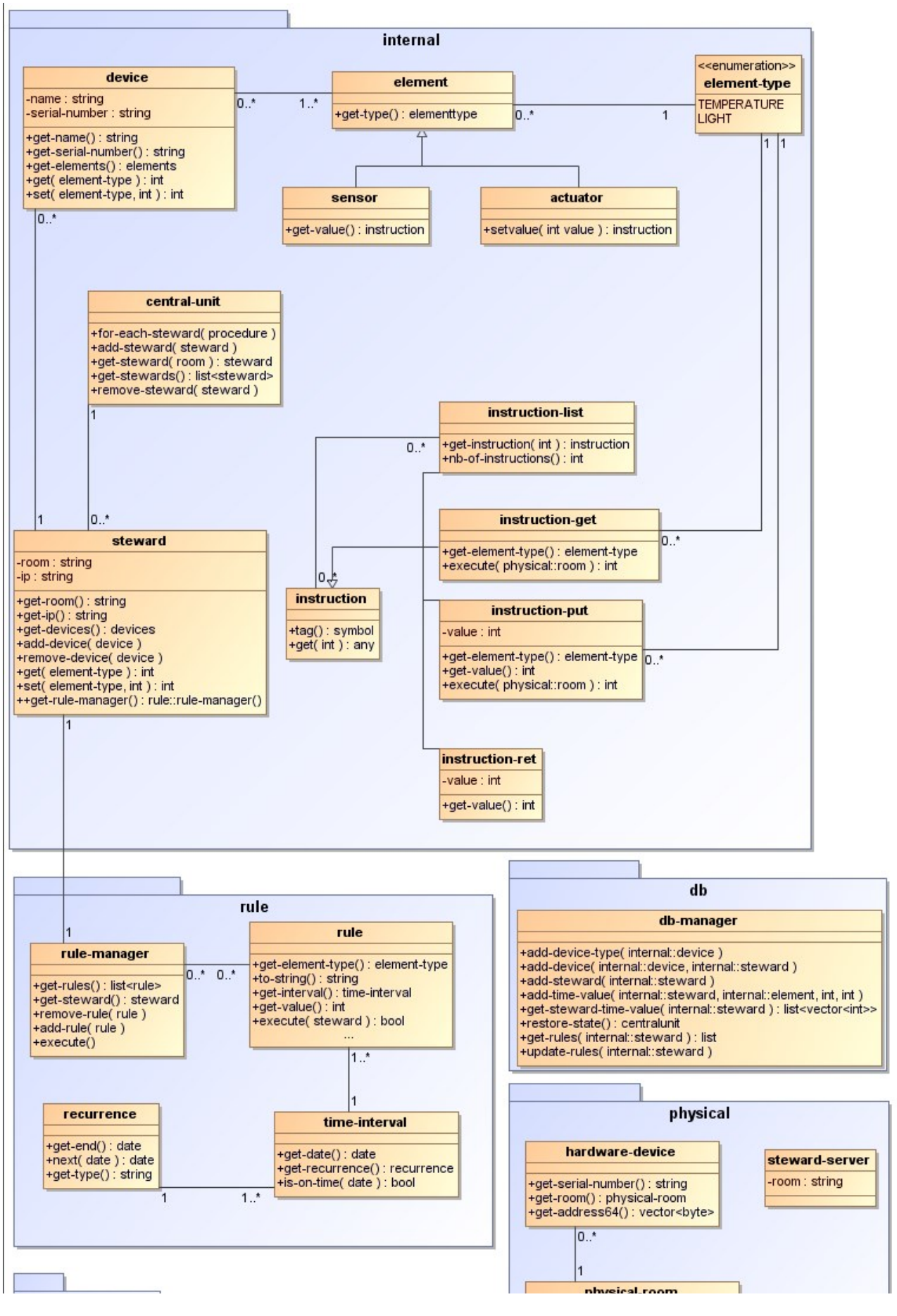
domotica

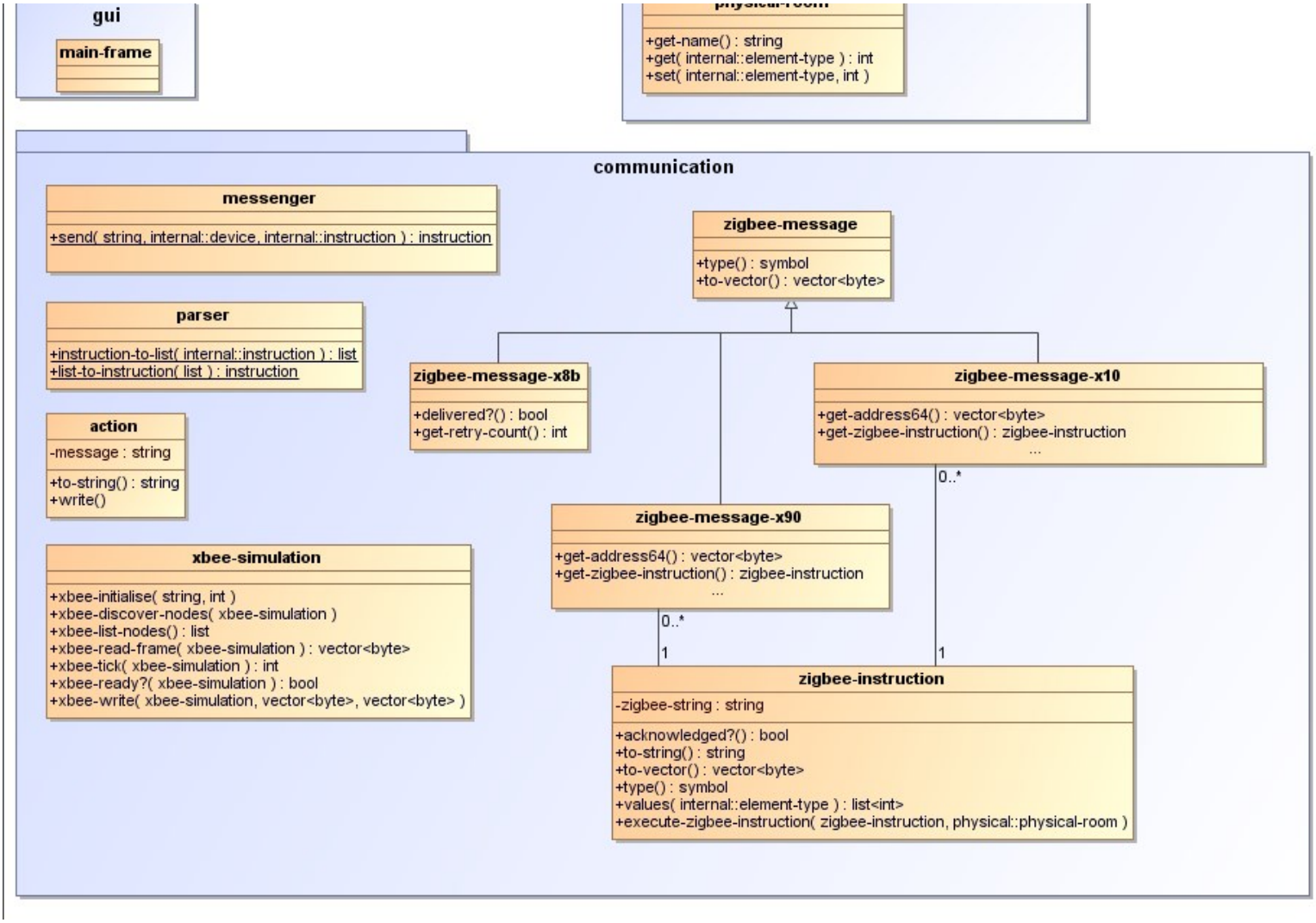
DESCRIPTION

Domotica is een systeem die het energieverbruik in een gebouw beheerst. Elke kamer van het gebouw bevat een steward die met zijn verschillende hardware toestellen communiceert. De verschillende stewards worden beheerst door een centrale server : de central-unit. De gebruiker kan via de central-unit de data van de stewards (temperatuur...) opvragen en aanpassen. Ook is het mogelijk om via een rule-systeem een planning op te stellen zodanig dat de data op een specifieke tijd vanzelf aangepast wordt.

Het **domotica** systeem is onderverdeeld in verschillende modules :

- internal - bevat de kern van het **domotica** systeem.
- communication - voorziet methodes om met de steward en hardware te communiceren.
- physical - simuleert het gedrag van de hardware en bevat de server klasse die op de steward moet runnen.
- rule - voorziet de verschillende klassen omtremt het rule-systeem.
- structure - extra ADT's die nodig zijn in het programma (zoals map).
- db - bevat de gegevensbank-manager.
- gui - hiermee kan de gebruiker interageren met de central-unit.
- unit-test - de klassen van de modules worden hierin getest.





1.1. domotica/communication [Modules]

NAME

communication

DESCRIPTION

Laat toe om boodschappen te sturen naar de hardware. Dit module bevat het nodige om te communiceren tussen de steward en de hardware door het zenden en opvangen van instructions. Berichten kunnen verzonden worden via de messenger, instructions die verzonden worden kunnen geparsed worden via de parser, het zigbee protocol kan geinterpreteerd worden door de zigbee-messages, en er kan naar de xbee geschreven worden (ook gesimuleerd) vie xbee(-simulation).rkt.

1.1.1. communication/action [Classes]

NAME

action

DESCRIPTION

De **action** classe schrijft messages naar een log file. Zo worden de instructions die een steward sendt in een log-file opgeslagen.

1.1.1.1. action/new-action [Constructors]

NAME

new-action

DESCRIPTION

Maakt een nieuw action object.

SYNOPSIS

```
25 (define (new-action message)
```

PARAMETERS

```
message - de string die geschreven moet worden.
```

1.1.1.2. action/to-string [Methods]

NAME

to-string

DESCRIPTION

Een string representatie van het action object.

SYNOPSIS

```
36     (define (action-to-string)
```

RETURN VALUE

```
string - de representatie van het action object.
```

1.1.1.3. action/write [Methods]

NAME

write

DESCRIPTION

Schrijft de action in de log file.

SYNOPSIS

```
48     (define (write)
```

RETURN VALUE

```
#<void>
```

1.1.2. communication/messenger [Classes]

NAME

messenger

DESCRIPTION

De **messenger** handelt de communicatie tussen de central-unit en de steward-server af. Instructies worden via tcp/ip verzonden, de steward-server voert de instruction dan uit en geeft een antwoord terug die door de **messenger** gelezen wordt.

1.1.2.1. messenger/send [Methods]

NAME

send

DESCRIPTION

Zendt een instructie naar een specifieke steward-server via een tcp port.

SYNOPSIS

```
52 (define (send steward-room device instruction)
```

PARAMETERS

```
* steward-room - de naam van de kamer van de ontvanger
* device - de device waarop de instructie toegepast moet worden.
* instruction - de instructie die toegepast moet worden.
```

RETURN VALUE

```
intsruction - het antwoord van de steward-server
```

1.1.3. communication/parser [Classes]

NAME

parser

DESCRIPTION

De **parser** zet instruction objecten om naar een lijst. Deze lijst kan dan verzonden worden via de messenger. Deze klasse kan ook een lijst terug omzetten naar een instruction object.

1.1.3.1. parser/instruction-to-list [Methods]

NAME

instruction-to-list

DESCRIPTION

Zet een instruction object om naar een list.

SYNOPSIS

```
27 (define (instruction-to-list instruction)
```

PARAMETERS

```
instruction - de instruction die omgezet moet worden.
```

RETURN VALUE

```
list - een lijst die de instruction voorstelt.
```

1.1.3.2. parser/list-to-instruction [Methods]

NAME

list-to-instruction

DESCRIPTION

Zet een lijst die een instructie voorstelt om naar een instruction object.

SYNOPSIS

```
52 (define (list-to-instruction lst)
```

PARAMETERS

```
lst - de list die omgezet moet worden.
```

RETURN VALUE

instruction - het overeenkomstige instruction object.

1.1.4. communication/steward-port-map [Variables]

NAME

steward-port-map

DESCRIPTION

Een map die de stewards en hun input en output ports bijhoudt. Elke tupel is (steward-ip >< (cons input-port output-port))

1.1.5. communication/xbee-simulation [Classes]

NAME

xbee-simulation

DESCRIPTION

Dit is een klasse die het gedrag van een xbee-device nadoet. De methodes van deze klasse zijn dezelfde als de methodes van de xbee library. Zigbee berichten worden op dezelfde manier behandelt als op een echt xbee toestel.

1.1.5.1. xbee-simulation/xbee-initialise [Constructors]

NAME

xbee-initialise

DESCRIPTION

Maakt een nieuw xbee object aan. Een xbee object heeft een buffer waarin de antwoorden van berichten opgeslagen worden. De berichten worden van de buffer overgeschreven naar een frame list, die dan door de gebruiker gelezen kan worden.

SYNOPSIS

```
46 (define (xbee-initialise port rate)
```

PARAMETERS

port - de port waarop het xbee device zich bevindt.
rate - standaart 9600

1.1.5.2. xbee-simulation/xbee-discover-nodes [Methods]

NAME

xbee-discover-nodes

DESCRIPTION

Deze methode zoekt naar hardware-devices waarmee hij kan communiceren en houdt ze in een list bij. Elke hardware-device in de list is voorgesteld als : (cons string-id 64bit-address) Om de list op te vragen wordt de methode xbee-list-nodes gebruikt.

SYNOPSIS

```
86 (define (xbee-discover-nodes xbee)
```

PARAMETERS

xbee - het xbee toetsel die naar de hardware gaat zoeken.

RETURN VALUE

```
#<void>
```

1.1.5.3. xbee-simulation/xbee-list-nodes [Methods]

NAME

xbee-list-nodes

DESCRIPTION

Geeft de list van gevonden hardware terug die opgezocht werd door xbee-discover-nodes.

SYNOPSIS

```
102 (define (xbee-list-nodes)
```

RETURN VALUE

```
list - de list van gevonden nodes
```

1.1.5.4. xbee-simulation/xbee-read-frame [Methods]

NAME

xbee-read-frame

DESCRIPTION

Leest de eerste frame van een xbee. Elke frame is een zigbee vector die via new-zigbee-message omgezet kan worden naar een zigbee-message object.

SYNOPSIS

```
149 (define (xbee-read-frame xbee)
```

PARAMETERS

```
xbee - de xbee waarvan de frame gelezen gaat worden.
```

RETURN VALUE

```
vector - een vector die een zigbee-message voorstelt.
```

1.1.5.5. xbee-simulation/xbee-ready? [Methods]

NAME

xbee-ready?

DESCRIPTION

Bekijkt of de buffer van een bepaalde xbee leeg is.

SYNOPSIS

```
132 (define (xbee-ready? xbee)
```

PARAMETERS

```
xbee - de xbee waarvan de buffer bekeken wordt.
```

RETURN VALUE

```
#t - true wanneer de buffer niet leeg is.
#f - false wanneer de buffer leeg is.
```

1.1.5.6. xbee-simulation/xbee-tick [Methods]

NAME

xbee-tick

DESCRIPTION

De buffer wordt door deze methode leeggemaakt zodat de zigbee berichten gelezen kunnen worden via xbee-read-frame.

SYNOPSIS

```
117 (define (xbee-tick xbee)
```

PARAMETERS

```
xbee - de xbee waarvan de buffer leeggemaakt gaat worden.
```

RETURN VALUE

```
#<void>
```

1.1.5.7. xbee-simulation/xbee-write [Methods]

NAME

xbee-write

DESCRIPTION

Schrijft een zigbee-message naar een xbee-device, die het bericht uitvoert. Eerst wordt er een bericht teruggeschreven naar de xbee die het delivery status van de message toont (van type zigbee-transmit-status of x8b). Daarna wordt het antwoord (van type zigbee-recieve-paquet of x90) geschreven in de buffer.

SYNOPSIS

```
172 (define (xbee-write xbee target message)
```

PARAMETERS

```
xbee - het toestel die de message gaat uitvoeren.
target - het 64bit adres van de hardware waarnaar het bericht verzonden moet worden.
message - een vector die een zigbee-message voorstelt.
```

RETURN VALUE

```
#<void>
#f - false als de hardware van het gegeven adres niet gevonden werd.
```

1.1.6. communication/zigbee-instruction [Classes]

NAME

zigbee-instruction

DESCRIPTION

Een class die een zigbee instructie voorstelt. De zigbee-string wordt naar een object omgezet zodanig dat de verschillende onderdelen van de string gemakkelijk opvraagbaar worden.

1.1.6.1. zigbee-instruction/new-zigbee-instruction [Constructors]

NAME

new-zigbee-instruction

DESCRIPTION

Maakt een nieuw zigbee-instruction object aan.

SYNOPSIS

```
229 (define (new-zigbee-instruction zigbee-string)
```

PARAMETERS

```
* zigbee-string - de zigbee-string die door deze instructie voorgesteld gaat worden.
```

1.1.6.2. zigbee-instruction/acknowledged? [Methods]

NAME

acknowledged?

DESCRIPTION

Is deze instructie een ACK instructie ? bv. "ACK: SET POW=ON\n" geeft #t terug, maar "GET\n" niet.

SYNOPSIS

```
280     (define (acknowledged?)
```

RETURN VALUE

```
vector - de bytevector
```

1.1.6.3. zigbee-instruction/execute-zigbee-instruction [Methods]

NAME

execute-zigbee-instruction

DESCRIPTION

Voer een zigbee-instruction uit. Dit wordt alleen gebruikt tijdens het simuleren van de hardware, omdat het xbee toestel normaalgezien de zigbee instructies vanzelf uitvoert.

SYNOPSIS

```
340 (define (execute-zigbee-instruction zigbee-instruction phys-room)
```

PARAMETERS

```
* zigbee-instruction - de zigbee-instruction die uitgevoerd moet worden.
* phys-room - de physical-room waarin de instructie uitgevoerd moet worden.
```

RETURN VALUE

```
zigbee-string - het gesimuleerde antwoord van de hardware : een string die het zigbee protocol volgt.
```

1.1.6.4. zigbee-instruction/to-string [Methods]

NAME

to-string

DESCRIPTION

geeft de een string terug die deze instructie voorstelt.

SYNOPSIS

```
255     (define (to-string)
```

RETURN VALUE

```
string - de zigbee string
```

1.1.6.5. zigbee-instruction/to-vector [Methods]

NAME

to-vector

DESCRIPTION

geeft de bytevector van de string voorstelling van deze instructie terug.

SYNOPSIS

```
267     (define (to-vector)
```

RETURN VALUE

```
vector - de bytevector
```

1.1.6.6. zigbee-instruction/type [Methods]

NAME

type

DESCRIPTION

Geeft het **type** terug van de zigbee-string bv. "ACK: SET POW=ON\n" geeft 'set terug, "GET\n" geeft 'get.

SYNOPSIS

```
295     (define (type)
```

RETURN VALUE

```
symbol - de symbol representatie van het type.
```

1.1.6.7. zigbee-instruction/values [Methods]

NAME

values

DESCRIPTION

Geeft het een lijst met de waarden van de zigbee-string voor een bepaald element-type terug. bv. "ACK: SET POW=ON\n" geeft (list 1) terug voor TEMPERATURE, "GET\n" geeft zowiezo '().

SYNOPSIS

```
311     (define (values element-type)
```

PARAMETERS

* element-type - het gevraagde element-type.

RETURN VALUE

list - lijst met waarden.

1.1.7. communication/zigbee-message [Classes]

NAME

zigbee-message

DESCRIPTION

Een classe die een zigbee bytevector voorstelt.

CHILDREN

* zigbee-message-x10
* zigbee-message-x8b
* zigbee-message-x90

1.1.7.1. zigbee-message/new-zigbee-message [Constructors]

NAME

new-zigbee-message

DESCRIPTION

Maakt een nieuw zigbee-message aan in functie van het meegegeven type. Dus voor type :

* zigbee-transmit-request wordt new-zigbee-message-x10 opgeroepen
* zigbee-transmit-status wordt new-zigbee-message-x8b opgeroepen
* zigbee-recieve-paquet wordt new-zigbee-message-x90 opgeroepen

Als er maar een vector meegegeven wordt als parameter wordt er veronderstelt dat deze een bytevector van een zigbee-message is. De eerste byte wordt gelezen en als type van de zigbee-message gezien. De vector wordt dan gesplitst en de correcte constructor opgeroepen.

SYNOPSIS

```
676 (define (new-zigbee-message type . args)
```

PARAMETERS

* type - het type van de message
* args - de argumenten van de verschillende zigbee-messages.
  In het geval dat er geen extra argumenten zijn wordt type gezien als een zigbee vector.

1.1.7.2. zigbee-message/zigbee-recieve-paquet [Variables]

NAME

zigbee-recieve-paquet

DESCRIPTION

Het type van het **zigbee-recieve-paquet** bericht. (zie zigbee-message-x90)

1.1.7.3. zigbee-message/zigbee-transmit-request [Variables]

NAME

zigbee-transmit-request

DESCRIPTION

Het type van het **zigbee-transmit-request** bericht. (zie zigbee-message-x10)

1.1.7.4. zigbee-message/zigbee-transmit-status [Variables]

NAME

zigbee-transmit-status

DESCRIPTION

Het type van het **zigbee-transmit-status** bericht. (zie zigbee-message-x8b)

1.1.7.5. zigbee-message/zigbee-vector-to-zigbee-string [Methods]

NAME

zigbee-vector-to-zigbee-string

DESCRIPTION

zet een zigbee bytevector om naar zijn string-representatie

SYNOPSIS

```
202 (define (zigbee-vector-to-zigbee-string zigbee-vector)
```

PARAMETERS

```
* zigbee-vector - de bytevector die omgezet moet worden
```

RETURN VALUE

```
string - een zigbee-string
```

1.1.8. communication/zigbee-message-x10 [Classes]

NAME

zigbee-message-x10

DESCRIPTION

Een classe die een zigbee bytevector van type x10 voorstelt. De opstelling van een zigbee vector (x10) :

```
* 0 : het type (in dit geval x10/16)
* 1 : de frame id
* 2-9 : 64bit adres
* 10-11 : 16bit adres
* 12 : broadcast-radius
* 13 : opties
* 14-* : de bytevector representatie van een zigbee string
```

PARENTS

```
* zigbee-message
```

1.1.8.1. zigbee-message-x10/new-zigbee-message-x10 [Constructors]

NAME

new-zigbee-message-x10

DESCRIPTION

Maakt een nieuw zigbee-message-x10 object aan.

SYNOPSIS

```
394 (define (new-zigbee-message-x10 id address64 address16 broadcast-radius options zigbee-vector)
```

PARAMETERS

```
* id - de id van het bericht
* address64 - het 64bit adres
* address16 - het 16bit adres
* broadcast-radius - de broadcast-radius
* options - de opties
* zigbee-vector - de bytevector representatie van een zigbee string
```

1.1.8.2. zigbee-message-x10/get-address64 [Methods]

NAME

get-address64

DESCRIPTION

Het 64 bit adres van deze zigbee instructie.

SYNOPSIS

```
436     (define (get-address64)
```

RETURN VALUE

```
vector - een bytevector met het 64 bit adres.
```

1.1.8.3. zigbee-message-x10/get-zigbee-instruction [Methods]

NAME

get-zigbee-instruction

DESCRIPTION

Geeft een zigbee-instruction object terug van deze message.

SYNOPSIS

```
448     (define (get-zigbee-instruction)
```

RETURN VALUE

```
zigbee-instruction - de instructie van dit bericht.
```

1.1.8.4. zigbee-message-x10/to-vector [Methods]

NAME

to-vector

DESCRIPTION

De bytevector representatie van deze zigbee instructie.

SYNOPSIS

```
424     (define (to-vector)
```

RETURN VALUE

```
vector - een bytevector die deze instructie voorstelt.
```

1.1.8.5. zigbee-message-x10/type [Methods]

NAME

type

DESCRIPTION

Geeft het **type** terug van de instructie (in dit geval zigbee-transmit-request).

SYNOPSIS

```
412     (define (type)
```

RETURN VALUE

```
integer - het type van deze instructie
```

1.1.9. communication/zigbee-message-x8b [Classes]

NAME

zigbee-message-x8b

DESCRIPTION

Een classe die een zigbee bytevector van type x8b voorstelt. De opstelling van een zigbee vector (x8b) :

```
* 0 : het type (in dit geval x8b/139)
* 1 : de frame id
* 2-3 : 16bit adres
* 4 : retry-count
* 5 : delivery-status (x00 is delivered, x24 is address not found)
* 6 : discovery-status
```

PARENTS

```
* zigbee-message
```

1.1.9.1. zigbee-message-x8b/new-zigbee-message-x8b [Constructors]

NAME

new-zigbee-message-x8b

DESCRIPTION

Maakt een nieuw zigbee-message-x8b object aan.

SYNOPSIS

```
490 (define (new-zigbee-message-x8b id address16 retry-count delivery-status discovery-status)
```

PARAMETERS

```
* id - de id van het bericht
* address16 - het 16bit adres
* retry-count - het aantal keren dat het bericht verzonden werd
* delivery-status - x00 is delivered, x24 is address not found
* discovery-status - het discovery-status
```

1.1.9.2. zigbee-message-x8b/delivered? [Methods]

NAME

delivered?

DESCRIPTION

Geeft aan of het bericht aangekomen is.

SYNOPSIS

```
531     (define (delivered?)
```

RETURN VALUE

```
#t - true wanneer het aangekomen is.
#f - false wanneer het niet aangekomen is.
```

1.1.9.3. zigbee-message-x8b/get-retry-count [Methods]

NAME

get-retry-count

DESCRIPTION

Geeft het aantal retries van het bericht.

SYNOPSIS

```
543     (define (get-retry-count)
```

RETURN VALUE

```
integer - het retry count.
```

1.1.9.4. zigbee-message-x8b/to-vector [Methods]

NAME

to-vector

DESCRIPTION

De bytevector representatie van deze zigbee instructie.

SYNOPSIS

```
518     (define (to-vector)
```

RETURN VALUE

```
vector - een bytevector die deze instructie voorstelt.
```

1.1.9.5. zigbee-message-x8b/type [Methods]

NAME

type

DESCRIPTION

Geeft het **type** terug van de instructie (in dit geval zigbee-transmit-status).

SYNOPSIS

```
506     (define (type)
```

RETURN VALUE

```
integer - het type van deze instructie
```

1.1.10. communication/zigbee-message-x90 [Classes]

NAME

zigbee-message-x90

DESCRIPTION

Een classe die een zigbee bytevector van type x90 voorstelt. De opstelling van een zigbee vector (x90) :

```
* 0 : het type (in dit geval x90/144)
* 1-8 : 64bit adres
* 9-10 : 16bit adres
* 11 : recieve option
* 12-* : de bytevector van een zigbee-instruction
```

PARENTS

```
* zigbee-message
```

1.1.10.1. zigbee-message-x90/new-zigbee-message-x90 [Constructors]

NAME

new-zigbee-message-x90

DESCRIPTION

Maakt een nieuw zigbee-message-x90 object aan.

SYNOPSIS

```
581 (define (new-zigbee-message-x90 address64 address16 recieve-option zigbee-vector)
```

PARAMETERS

```
* address64 - het 64bit adres
* address16 - het 16bit adres
* recieve-option - de recieve-option
* zigbee-vector - de bytevector van een zigbee-string
```

1.1.10.2. zigbee-message-x90/get-address64 [Methods]

NAME

get-address64

DESCRIPTION

Het 64bit adres van de ontvanger van dit bericht.

SYNOPSIS

```
622     (define (get-address64)
```

RETURN VALUE

vector - een vector met het 64bit adres van de ontvanger

1.1.10.3. zigbee-message-x90/get-zigbee-instruction [Methods]

NAME

get-zigbee-instruction

DESCRIPTION

Geeft een zigbee-instruction object terug van deze message.

SYNOPSIS

```
634     (define (get-zigbee-instruction)
```

RETURN VALUE

zigbee-instruction - de instructie van dit bericht.

1.1.10.4. zigbee-message-x90/to-vector [Methods]

NAME

to-vector

DESCRIPTION

De bytevector representatie van deze zigbee instructie.

SYNOPSIS

```
610     (define (to-vector)
```

RETURN VALUE

vector - een bytevector die deze instructie voorstelt.

1.1.10.5. zigbee-message-x90/type [Methods]

NAME

type

DESCRIPTION

Geeft het **type** terug van de instructie (in dit geval zigbee-recieve-paquet).

SYNOPSIS

```
598     (define (type)
```

RETURN VALUE

integer - het **type** van deze instructie

1.2. domotica/db [Modules]

NAME

db

DESCRIPTION

Beheerst de gegevensbank van het systeem. Elke steward en device die door de user aangemaakt wordt wordt opgeslagen in de database. De gegevensbank :

- De Stewards table laat toe om stewards op te slaan.
- De Devices table laat toe om devices op te slaan.
- De ElementTypes table toont welke types meetbaar en/of beinvloedbaar kunnen zijn door sensoren en actuatoren.
- De DeviceSensors table toont, voor elk type device (devices zijn van hetzelfde type wanneer ze dezelfde naam hebben), welke sensoren het heeft.
- De DeviceActuators table toont, vool elk type device (devices zijn van hetzelfde type wanneer ze dezelfde naam hebben), welke actuatoren het heeft.
- De ElementTypeValues table toont de waarde van een bepaalde element-type in een bepaalde kamer op een bepaalde tijd.

Name	Object	Type
Stewards	table	
room	field	char(20) PRIMARY KEY
ip	field	char(20)
Devices	table	
serial	field	char(20) PRIMARY KEY
name	field	char(20)
steward	field	char(20)
ElementTypes	table	
type	field	int PRIMARY KEY
sensor	field	bool
actuator	field	bool
DeviceSensors	table	
device	field	char(20) PRIMARY KEY
element	field	int PRIMARY KEY
DeviceActuators	table	
device	field	char(20) PRIMARY KEY
element	field	int PRIMARY KEY
ElementTypeValues	table	
room	field	char(20)
element	field	int
time	field	int
value	field	int
Rules	table	
room	field	char(20) PRIMARY KEY
time	field	int PRIMARY KEY
element	field	int PRIMARY KEY
value	field	int
recurrence	field	char(20)
end	field	int

1.2.1. db/db-manager [Classes]

NAME

db-manager

DESCRIPTION

Een manager die toelaat om gemakkelijk te interageren met de gegevensbank. De manager slaagt de verschillende devices, stewards en resultaten van instructions op.

1.2.1.1. db-manager/new-db-manager [Constructors]

NAME

new-db-manager

DESCRIPTION

maakt een nieuwe db-manager aan gebaseerd op de database die op db-path ligt.

SYNOPSIS

```
101 (define (new-db-manager db-path)
```

PARAMETERS

* db-path - het pad van de gegevensbank.

1.2.1.2. db-manager/add-device [Methods]

NAME

add-device

DESCRIPTION

Voeg een device toe aan de gegevensbank.

SYNOPSIS

```
154   (define (add-device device steward)
```

PARAMETERS

* device - de device die toegevoegd moet worden.
* steward - de steward die het device beheerst.

RETURN VALUE

#<void>

1.2.1.3. db-manager/add-device-type [Methods]

NAME

add-device-type

DESCRIPTION

Voeg een nieuw device-type toe aan de database.

SYNOPSIS

```
116   (define (add-device-type device)
```

PARAMETERS

* device - een device van het type die toegevoegd moet worden.

RETURN VALUE

#<void>

1.2.1.4. db-manager/add-steward [Methods]

NAME

add-steward

DESCRIPTION

Voeg een steward toe aan de gegevensbank.

SYNOPSIS

```
251   (define (add-steward steward)
```

PARAMETERS

* steward - de steward die toegevoegd moet worden.

RETURN VALUE

```
#<void>
```

1.2.1.5. db-manager/add-time-value [Methods]

NAME

add-time-value

DESCRIPTION

Slaagt de waarde van een bepaalde element-type in een bepaalde kamer op een bepaald tijdstip op in de gegevensbank.

SYNOPSIS

```
292   (define (add-time-value steward element time value)
```

PARAMETERS

```
* steward - de steward die het element-type gemeten heeft.
* element - het element-type die gemeten werd.
* time - het tijdstip (in seconden) van de meting.
* value - de waarde van de meting.
```

RETURN VALUE

```
#<void>
```

1.2.1.6. db-manager/get-rules [Methods]

NAME

get-rules

DESCRIPTION

Geeft een list terug met alle rules die op een steward toegepast zijn.

SYNOPSIS

```
198   (define (get-rules steward)
```

PARAMETERS

```
* steward - de steward van wie we de rules willen krijgen
```

RETURN VALUE

```
* list - een lijst met de rules van de steward
```

1.2.1.7. db-manager/remove-device [Methods]

NAME

remove-device

DESCRIPTION

Verwijdert een device van de gegevensbank.

SYNOPSIS

```
178   (define (remove-device device)
```

PARAMETERS

* device - de device die verwijdert moet worden.

RETURN VALUE

#<void>

1.2.1.8. db-manager/restore-state [Methods]

NAME

restore-state

DESCRIPTION

Geeft een central-unit terug met de huidige stand van de gegevensbank. De central-unit beheerst dus alle verschillende opgeslagen stewards en opgeslagen devices.

SYNOPSIS

```
336   (define (restore-state)
```

RETURN VALUE

central-unit - een central-unit met de huidige stand van de database.

1.2.1.9. db-manager/update-rules [Methods]

NAME

update-rules

DESCRIPTION

Update de rules van een steward in de database.

SYNOPSIS

```
226   (define (update-rules steward)
```

PARAMETERS

* steward - de steward van wie we de rules willen updaten.

RETURN VALUE

#<void>

1.3. domotica/gui [Modules]

NAME

gui

DESCRIPTION

Het **gui** module bevat de klassen omtremt de Graphical User Interface. Hiermee kan de gebruiker interageren met het programma en aan de central-unit vragen om stewards of devices toe te voegen of verwijderen. Hij kan nieuwe rules maken en instructions zenden naar de hardware.

1.4. domotica/internal [Modules]

NAME

internal

DESCRIPTION

Het **internal** module bevat de kern van het domotica systeem. De central-unit houdt de verschillende stewards bij, die instructies kunnen zenden aan de hardware, zoals het aanpassen van de licht-intensiteit of het opvragen van de temperatuur van de kamer.

1.4.1. internal/actuator [Classes]

NAME

actuator

DESCRIPTION

Laat toe om de informatie van zijn element-type aan te passen. Een **actuator** genereert een instruction die later verzonden kan worden naar de hardware. Deze instruction past de waarde van element-type naar een nieuwe waarde toe.

PARENTS

* element

1.4.1.1. actuator/new-actuator [Constructors]

NAME

new-actuator

DESCRIPTION

Maakt een nieuwe actuator aan.

SYNOPSIS

```
36 (define (new-actuator element-type)
```

PARAMETERS

* element-type - het element-type van de actuator.

1.4.1.2. actuator/class [Methods]

NAME

class

DESCRIPTION

Geeft de classe terug van dit object.

SYNOPSIS

```
48     (define (class)
```

RETURN VALUE

symbol - de naam van de classe

1.4.1.3. actuator/set-value [Methods]

NAME

set-value

DESCRIPTION

Geeft een instruction-put terug die het element-type van deze actuator aanpast op een bepaalde waarde.

SYNOPSIS

```
79      (define (set-value value)
```

PARAMETERS

```
* value - De nieuwe waarde.
```

RETURN VALUE

```
instruction-put - de gevraagde instructie.
```

1.4.1.4. actuator/super [Methods]

NAME

super

DESCRIPTION

Geeft de **super**-classe van dit object terug

SYNOPSIS

```
61      (define (super)
```

RETURN VALUE

```
element - de super-classe
```

1.4.2. internal/central-unit [Classes]

NAME

central-unit

DESCRIPTION

De **central-unit** bevat de verschillende stewards van het domotica systeem. Alle interactie die met de gebruiker via de GUI gebeurt wordt in de achtergrond door de **central-unit** en zijn stewards afgehandeld.

1.4.2.1. central-unit/new-central-unit [Constructors]

NAME

new-central-unit

DESCRIPTION

Maakt een nieuwe central-unit aan.

SYNOPSIS

```
26 (define (new-central-unit)
```

1.4.2.2. central-unit/add-steward [Methods]

NAME

add-steward

DESCRIPTION

Voeg een steward toe.

SYNOPSIS

```
77      (define (add-steward steward)
```

PARAMETERS

steward - steward die toegevoegd moet worden.

RETURN VALUE

#<void>

1.4.2.3. central-unit/class [Methods]

NAME

class

DESCRIPTION

Geeft de classe terug van dit object.

SYNOPSIS

```
37      (define (class)
```

RETURN VALUE

symbol - de naam van de classe

1.4.2.4. central-unit/for-each-steward [Methods]

NAME

for-each-steward

DESCRIPTION

Past een procedure toe op elke steward van deze central-unit.

SYNOPSIS

```
63      (define (for-each-steward proc)
```

PARAMETERS

proc - de procedure die toegepast gaat worden.

RETURN VALUE

#<void>

1.4.2.5. central-unit/get-steward [Methods]

NAME

get-steward

DESCRIPTION

Zoek naar de steward van een bepaalde kamer.

SYNOPSIS

```
95      (define (get-steward room)
```

PARAMETERS

```
room - de kamer van de gezochte steward.
```

RETURN VALUE

```
steward - de gevonden steward.
#f - false wanneer er geen overeenkomstige steward is.
```

1.4.2.6. central-unit/get-stewards [Methods]

NAME

get-stewards

DESCRIPTION

Geeft de lijst van stewards die deze central-unit beheerst terug.

SYNOPSIS

```
49      (define (get-stewards)
```

RETURN VALUE

```
list - de stewards van deze central-unit.
```

1.4.2.7. central-unit/remove-steward [Methods]

NAME

remove-steward

DESCRIPTION

Verwijder een specifieke steward. Deze wordt niet langer beheerst door de central-unit.

SYNOPSIS

```
111     (define (remove-steward steward)
```

PARAMETERS

```
steward - de steward die verwijdert moet worden.
```

RETURN VALUE

```
#<void> - wanneer de steward verwijderd werd.
#f - false wanneer er geen overeenkomstige steward is.
```

1.4.3. internal/device [Classes]

NAME

device

DESCRIPTION

Een toestel die een aantal sensoren en actuatoren bevat. Een **device** kan de element-types van de kamer waarin hij zich bevindt

opvragen en aanpassen, vermits hij bestaat uit de corresponderende sensoren en actuatoren. Elke **device** heeft een naam en een unieke identifier. Elke **device** die dezelfde sensoren en actuatoren bevat is beschouwd als hetzelfde type **device**, en devices die van hetzelfde type zijn hebben ook dezelfde naam. Dus, wanneer een nieuw **device** aangemaakt wordt en zijn naam hetzelfde is als een van de bestaande devices wordt er automatisch de corresponderende sensoren en actuatoren toegevoegd.

1.4.3.1. device/new-device [Constructors]

NAME

new-device

DESCRIPTION

Maakt een nieuw device aan.

SYNOPSIS

```
83 (define (new-device name serial-number)
```

PARAMETERS

```
* name - de naam van het device. Elke device met dezelfde naam is van hetzelfde type.
* serial-number - de unieke identifier van deze device.
```

1.4.3.2. device/add-element [Methods]

NAME

add-element

DESCRIPTION

Voeg een nieuw element toe aan deze device.

SYNOPSIS

```
144     (define (add-element element)
```

PARAMETERS

```
* element - het element die toegevoegd moet worden.
```

RETURN VALUE

```
#<void>
```

1.4.3.3. device/class [Methods]

NAME

class

DESCRIPTION

Geeft de classe terug van dit object.

SYNOPSIS

```
94     (define (class)
```

RETURN VALUE

```
symbol - de naam van de classe
```

1.4.3.4. device/device-types [Variables]

NAME

device-types

DESCRIPTION

Deze variabele houdt verschillende type devices bij.

1.4.3.5. device/get [Methods]

NAME

get

DESCRIPTION

Vraagt informatie over het meegegeven element-type. De device zoekt naar een sensor met het corresponderende element-type, en geeft dan een instruction-get terug.

SYNOPSIS

```
161     (define (get element-type)
```

PARAMETERS

```
* element-type - het element-type waarvan de waarde gekend wilt worden.
```

RETURN VALUE

```
* instruction-get - een instructie als het de gevraagde sensor bevat
* #f - false als er geen sensor gevonden werd.
```

1.4.3.6. device/get-elements [Methods]

NAME

get-elements

DESCRIPTION

Geeft een lijst terug met de elements van deze device.

SYNOPSIS

```
130     (define (get-elements)
```

RETURN VALUE

```
list - Een lijst van de elements die de device bevat.
```

1.4.3.7. device/get-name [Methods]

NAME

get-name

DESCRIPTION

Geeft de naam van deze device terug.

SYNOPSIS

```
106     (define (get-name)
```

RETURN VALUE

string - de naam van deze device.

1.4.3.8. device/get-serial-number [Methods]

NAME

get-serial-number

DESCRIPTION

Geeft de unieke identifier van deze device terug.

SYNOPSIS

```
118     (define (get-serial-number)
```

RETURN VALUE

string - de unieke identifier van deze device.

1.4.3.9. device/set [Methods]

NAME

set

DESCRIPTION

Past de waarde van het meegegeven element-type aan. De device zoekt naar een actuator met het corresponderende element-type en geeft dan een instruction-put terug.

SYNOPSIS

```
190     (define (set element-type value)
```

PARAMETERS

* element-type - het element-type waarvan de waarde aangepast wilt worden.
* value - de waarde waarop het element-type gezet wilt worden.

RETURN VALUE

* instruction-put - een instructie als het de gevraagde actuator bevat.
* #f - false als er geen actuator gevonden werd.

1.4.4. internal/element [Classes]

NAME

element

DESCRIPTION

Een abstracte classe die een element-type bijhoudt.

CHILDREN

* sensor
* actuator

1.4.4.1. element/new-element [Constructors]

NAME

new-element

DESCRIPTION

Maakt een nieuw element object.

SYNOPSIS

```
35 (define (new-element element-type)
```

PARAMETERS

```
* element-type - het element-type die dit element representeert
```

1.4.4.2. element/class [Methods]

NAME

class

DESCRIPTION

Geeft de classe terug van dit object.

SYNOPSIS

```
45   (define (class)
```

RETURN VALUE

```
symbol - de naam van de classe
```

1.4.4.3. element/get-type [Methods]

NAME

get-type

DESCRIPTION

Geeft het element-type van dit element terug.

RETURN VALUE

```
element-type - het object's element-type
SYNOPSYS
```

1.4.5. internal/element-type [Classes]

NAME

element-type

DESCRIPTION

Deze classe stelt alle verschillende types elementen (zoals temperatuur) die door het domotica systeem gebruikt kunnen worden. Andere element-types die de verschillende sensoren kunnen meten zullen door het domotica-systeem genegeerd worden.

1.4.5.1. element-type/element-type-zigbee-type-map [Variables]

NAME

element-type-zigbee-type-map

DESCRIPTION

Elk element-type gebruikt door het domotica systeem heeft een specifieke representatie in het zigbee protocol. Deze map mapt elk element-type naar zijn zigbee representatie.

1.4.5.2. element-type/for-each-element-type [Methods]

NAME

for-each-element-type

DESCRIPTION

Deze methode past een procedure op elk element-type toe.

PARAMETERS

```
* proc - de procedure die op elk element-type toegepast moet worden.
```

RETURN VALUE

```
#<void>
SYNOPSYS
```

1.4.5.3. element-type/LIGHT [Variables]

NAME

LIGHT

DESCRIPTION

Stelt het licht voor.

1.4.5.4. element-type/TEMPERATURE [Variables]

NAME

TEMPERATURE

DESCRIPTION

Stelt de temperatuur voor.

1.4.5.5. element-type/to-string [Methods]

NAME

to-string

DESCRIPTION

De string-representatie van een element-type

SYNOPSIS

```
62 (define (to-string element-type)
```

PARAMETERS

```
* element-type - het type dat omgezet moet worden.
```

RETURN VALUE

string - de string-representatie van element-type

1.4.6. internal/instruction [Classes]

NAME

instruction

DESCRIPTION

Een abstracte classe die een instructie voor een bepaald element-type voorstelt. Instructions kunnen geparsed worden zodanig dat ze via een messenger verzonden kunnen worden.

CHILDREN

* instruction-get
* instruction-put
* instruction-ret

1.4.6.1. instruction/new-instruction [Constructors]

NAME

new-instruction

DESCRIPTION

Maakt een nieuwe instrucion aan.

SYNOPSIS

```
45 (define (new-instruction tag . args)
```

PARAMETERS

* tag - het type van de instructie
* args - een lijst van optionele argumenten

1.4.6.2. instruction/get [Methods]

NAME

get

DESCRIPTION

Geeft het i-de optionele argument terug.

SYNOPSIS

```
69   (define (get i)
```

PARAMETERS

* i - de positie van het nodige argument

RETURN VALUE

any - het i-de argument.

1.4.6.3. instruction/get-tag [Methods]

NAME

get-tag

DESCRIPTION

Geeft de tag van deze instruction terug.

SYNOPSIS

```
55   (define (get-tag)
```

RETURN VALUE

```
symbol - de tag van deze instruction.
```

1.4.7. internal/instruction-get [Classes]

NAME

instruction-get

DESCRIPTION

Instructie om de waarde van een bepaalde element-type te krijgen.

PARENTS

```
* instruction
```

1.4.7.1. instruction-get/new-instruction-get [Constructors]

NAME

new-instruction-get

DESCRIPTION

Maakt een nieuwe instrucion-get aan.

SYNOPSIS

```
101 (define (new-instruction-get element-type)
```

PARAMETERS

```
* element-type - het element-type van deze instruction.
```

1.4.7.2. instruction-get/execute [Methods]

NAME

execute

DESCRIPTION

Past de instruction toe op een device. De instruction schrijft het nodige zigbee bericht op de xbee.

SYNOPSIS

```
141     (define (execute xbee device-serial)
```

PARAMETERS

```
* xbee - de xbee waarop het bericht geschreven gaat worden.
```

* device-serial - het unieke identifier van de device.

RETURN VALUE

#<void> - wanneer het bericht verzonden werd.
#f - wanneer de xbee de meegegeven device niet vindt.

1.4.7.3. instruction-get/get-element-type [Methods]

NAME

get-element-type

DESCRIPTION

Geeft het **get-element-type** van deze instruction terug.

SYNOPSIS

```
124     (define (get-element-type)
```

RETURN VALUE

element-type - het element-type van deze instruction.

1.4.7.4. instruction-get/tag [Methods]

NAME

tag

DESCRIPTION

Geeft de **tag** van deze instruction terug.

SYNOPSIS

```
112     (define (tag)
```

RETURN VALUE

symbol - de **tag** van deze instruction.

1.4.7.5. instruction-get/value-of [Methods]

NAME

value-of

DESCRIPTION

Geeft de waarde van een zigbee-instruction terug die overeenkomt met deze instructie (bv. de waarde van POW als het element-type LIGHT was).

SYNOPSIS

```
175     (define (value-of zigbee-instruction)
```

PARAMETERS

* zigbee-instruction - de instructie die opgevraagd gaat worden

RETURN VALUE

integer - wanneer de instructie het gevraagde element-type bevat.
#f - wanneer de instructie niet overeenkomt met het element-type.

1.4.8. internal/instruction-put [Classes]

NAME

instruction-put

DESCRIPTION

Instructie om de waarde van een bepaalde element-type aan te passen.

PARENTS

* instruction

1.4.8.1. instruction-put/new-instruction-put [Constructors]

NAME

new-instruction-put

DESCRIPTION

Maakt een nieuwe instrucion-put aan.

SYNOPSIS

```
214 (define (new-instruction-put element-type value)
```

PARAMETERS

* element-type - het element-type van deze instruction.
* value - de waarde waarop het element-type gezet moet worden.

1.4.8.2. instruction-put/execute [Methods]

NAME

execute

DESCRIPTION

Past de instruction toe op een device. De instruction schrijft het nodige zigbee bericht op de xbee.

SYNOPSIS

```
266     (define (execute xbee device-serial)
```

PARAMETERS

* xbee - de xbee waarop het bericht geschreven gaat worden.
* device-serial - het unieke identifier van de device.

RETURN VALUE

#<void> - wanneer het bericht verzonden werd.
#f - wanneer de xbee de meegegeven device niet vindt.

1.4.8.3. instruction-put/get-element-type [Methods]

NAME

get-element-type

DESCRIPTION

Geeft het **get-element-type** van deze instruction terug.

SYNOPSIS

```
237     (define (get-element-type)
```

RETURN VALUE

element-type - het element-type van deze instruction.

1.4.8.4. instruction-put/get-value [Methods]

NAME

get-value

DESCRIPTION

Geeft de waarde van deze instruction terug.

SYNOPSIS

```
249     (define (get-value)
```

RETURN VALUE

integer - waarde van deze instruction

1.4.8.5. instruction-put/tag [Methods]

NAME

tag

DESCRIPTION

Geeft de **tag** van deze instruction terug.

SYNOPSIS

```
225     (define (tag)
```

RETURN VALUE

symbol - de **tag** van deze instruction.

1.4.8.6. instruction-put/value-of [Methods]

NAME

value-of

DESCRIPTION

Bekijkt of een zigbee-instruction correct werd uitgevoerd

SYNOPSIS

```
306     (define (value-of zigbee-instruction)
```

PARAMETERS

* zigbee-instruction - de instructie die opgevraagd gaat worden

RETURN VALUE

#t - de instructie werd correct uitgevoerd.
#f - de instructie werd niet correct uitgevoerd.

1.4.9. internal/instruction-ret [Classes]

NAME

instruction-ret

DESCRIPTION

Deze instructie wordt van de steward-server zijn overeenkomstige steward (in de central-unit) verzonden als antwoord op de berichten van de central-unit.

PARENTS

* instruction

1.4.9.1. instruction-ret/new-instruction-ret [Constructors]

NAME

new-instruction-ret

DESCRIPTION

Maakt een nieuwe instrucion-ret aan.

SYNOPSIS

```
362 (define (new-instruction-ret value)
```

PARAMETERS

* value - de waarde die naar de central-unit verzonden moet worden.

1.4.9.2. instruction-ret/get-value [Methods]

NAME

get-value

DESCRIPTION

Geeft de waarde van deze instruction terug.

SYNOPSIS

```
385     (define (get-value)
```

RETURN VALUE

(or integer #t #f) - waarde van deze instruction

1.4.9.3. instruction-ret/tag [Methods]

NAME

tag

DESCRIPTION

Geeft de **tag** van deze instruction terug.

SYNOPSIS

```
373     (define (tag)
```

RETURN VALUE

symbol - de **tag** van deze instruction.

1.4.10. internal/sensor [Classes]

NAME

sensor

DESCRIPTION

Vraagt informatie over zijn element-type op. Een **sensor** genereert een instruction die de waarde van zijn element-type opvraagt. Deze instruction kan daarna via een messenger verzonden worden naar hardware zodat die het element-type meet.

PARENTS

* element

1.4.10.1. sensor/new-sensor [Constructors]

NAME

new-sensor

DESCRIPTION

Maakt een nieuwe sensor aan.

SYNOPSIS

```
37 (define (new-sensor element-type)
```

PARAMETERS

* element-type - het element-type van de sensor.

1.4.10.2. sensor/class [Methods]

NAME

class

DESCRIPTION

Geeft de classe terug van dit object.

SYNOPSIS

```
49     (define (class)
```

RETURN VALUE

symbol - de naam van de classe

1.4.10.3. sensor/get-value [Methods]

NAME

get-value

DESCRIPTION

Geeft een instruction-get terug die de waarde van deze sensor's element-type kan opvragen.

SYNOPSIS

```
77      (define (get-value)
```

RETURN VALUE

instruction-get - de gevraagde instructie.

1.4.10.4. sensor/super [Methods]

NAME

super

DESCRIPTION

Geeft de **super**-classe van dit object terug

SYNOPSIS

```
62      (define (super)
```

RETURN VALUE

element - de **super**-classe

1.4.11. internal/steward [Classes]

NAME

steward

DESCRIPTION

Een **steward** bevindt zich in een kamer en heeft een uniek ip adres. Elke kamer bevat maximum een **steward**. Een **steward** beheerst een aantal devices en kan instructions zenden naar de hardware door gebruik te maken van een messenger. Een **steward** kan bepaalde rules hebben die automatisch instructions zenden wanneer de rule waar is.

1.4.11.1. steward/new-steward [Constructors]

NAME

new-steward

DESCRIPTION

Maakt een nieuwe steward aan.

SYNOPSIS

```
37 (define (new-steward room ip)
```

PARAMETERS

* room - de naam van de kamer waar de steward zich bevindt. Elke steward is in een verschillende kamer.
* ip - het ip adres van deze steward.

1.4.11.2. steward/add-device [Methods]

NAME

add-device

DESCRIPTION

Voeg een device toe aan deze steward.

SYNOPSIS

```
128     (define (add-device device)
```

PARAMETERS

* device - de device die toegevoegd moet worden.

RETURN VALUE

#<void>

1.4.11.3. steward/class [Methods]

NAME

class

DESCRIPTION

Geeft de classe terug van dit object.

SYNOPSIS

```
49      (define (class)
```

RETURN VALUE

symbol - de naam van de classe

1.4.11.4. steward/get [Methods]

NAME

get

DESCRIPTION

Zendt een instruction naar de hardware van een beheerste device via een messenger. Geeft het antwoord van de harware terug.

SYNOPSIS

```
149     (define (get element-type)
```

PARAMETERS

* element-type - het element-type die opgevraagd gaat worden.

RETURN VALUE

integer - de waarde die de hardware gemeten heeft.
#f - wanneer steward geen device beheerst die het gegeven element-type kan meten.

1.4.11.5. steward/get-devices [Methods]

NAME

get-devices

DESCRIPTION

Een lijst van de devices die deze steward beheert.

SYNOPSIS

```
85      (define (get-devices)
```

RETURN VALUE

```
list - de lijst met de beheerste devices.
```

1.4.11.6. steward/get-ip [Methods]

NAME

get-ip

DESCRIPTION

Geeft het ip adres van deze steward terug.

SYNOPSIS

```
73      (define (get-ip)
```

RETURN VALUE

```
string - het ip adres van deze steward.
```

1.4.11.7. steward/get-room [Methods]

NAME

get-room

DESCRIPTION

Geeft de naam van de kamer waar de steward zich bevindt terug.

SYNOPSIS

```
61      (define (get-room)
```

RETURN VALUE

```
string - de naam van de kamer.
```

1.4.11.8. steward/get-rule-manager [Methods]

NAME

get-rule-manager

DESCRIPTION

Geeft de rule-manager van deze steward terug.

SYNOPSIS

```
199     (define (get-rule-manager)
```

RETURN VALUE

rule-manager - deze steward's rule manager.

1.4.11.9. steward/remove-device [Methods]

NAME

remove-device

DESCRIPTION

Verwijdert een bepaalde device van de beheerste devices.

SYNOPSIS

```
100     (define (remove-device device)
```

PARAMETERS

* device - de device die verwijdert moet worden.

RETURN VALUE

#<void> - als de device verwijderd werd.
#f - wanneer de device niet door deze steward beheerst is.

1.4.11.10. steward/set [Methods]

NAME

set

DESCRIPTION

Zendt een instruction naar de hardware van een beheerste device via een messenger. Geeft het antwoord van de harware terug.

SYNOPSIS

```
173     (define (set element-type value)
```

PARAMETERS

* element-type - het element-type die aangepast moet worden.
* value - de waarde waarop element-type aangepast moet worden.

RETURN VALUE

#t - wanneer de waarde correct werd aangepast.
#f - wanneer de waarde niet werd aangepast.
#f - wanneer steward geen device beheerst die het gegeven element-type kan aanpassen.

1.5. domotica/physical [Modules]

NAME

physical

DESCRIPTION

Dit module bevat de verschillende klasses die gebruikt worden door de hardware of de hardware simuleren. De steward-server klasse runt op elke steward, terwijl hardware-device en physical-room gebruikt worden tijdens het simuleren van het domotica-systeem.

1.5.1. physical/hardware-device [Classes]

NAME

hardware-device

DESCRIPTION

Deze klasse wordt gebruikt voor het simuleren van het domotica systeem (via xbee-simulation). Het vervangt de echte hardware waarmee de xbee-device van een kamer communiceert.

1.5.1.1. hardware-device/new-hardware-device [Constructors]

NAME

new-hardware-device

DESCRIPTION

Maakt een nieuwe hardware-device object aan. Een hardware device bevindt zich in een bepaalde physical-room en heeft een uniek identifier.

SYNOPSIS

```
49 (define (new-hardware-device serial-number room)
```

PARAMETERS

```
* serial-number - het uniek identifier.
* room - de physical-room waarin deze physical-device zicht bevindt.
```

1.5.1.2. hardware-device/get-address64 [Methods]

NAME

get-address64

DESCRIPTION

Geeft het 64bit adres van deze hardware-device terug.

SYNOPSIS

```
72      (define (get-address64)
```

RETURN VALUE

```
vector - het 64bit adres.
```

1.5.1.3. hardware-device/get-room [Methods]

NAME

get-room

DESCRIPTION

Geeft de physical-room waarin deze hardware-device zich bevindt terug.

SYNOPSIS

```
84      (define (get-room)
```

RETURN VALUE

physical-room - de physical-room van deze hardware-device.

1.5.1.4. hardware-device/get-serial-number [Methods]

NAME

get-serial-number

DESCRIPTION

Geeft de unieke identifier van deze hardware device terug.

SYNOPSIS

```
60      (define (get-serial-number)
```

RETURN VALUE

string - de unieke identifier

1.5.1.5. hardware-device/hardware-device-map [Variables]

NAME

hardware-device-map

DESCRIPTION

Een map die alle aangemaakte hardware-devices bijhoudt. elke tupel is (serial-number >< hardware-device).

1.5.2. physical/physical-room [Classes]

NAME

physical-room

DESCRIPTION

Simuleert een kamer. Een kamer heeft verschillende element-type attributen zoals LIGHT. Deze klasse wordt gebruikt voor het simuleren van xbee toestellen (via xbee-simulation). Een zigbee-message wordt afgehandelt door execute-zigbee-instruction, die de attributen van **physical-room** gaat opvragen en aanpassen.

1.5.2.1. physical-room/new-physical-room [Constructors]

NAME

new-physical-room

DESCRIPTION

Maakt een nieuwe physical-room aan.

SYNOPSIS

```
43 (define (new-physical-room name)
```

PARAMETERS

* de naam van de physical-room.

1.5.2.2. physical-room/get [Methods]

NAME

get

DESCRIPTION

Geeft de waarde van een element-type van deze physical-room terug.

SYNOPSIS

```
68      (define (get element-type)
```

PARAMETERS

```
* element-type - de element-type waarvan de waarde gevraagd is.
```

RETURN VALUE

```
integer - de waarde van het element-type
```

1.5.2.3. physical-room/get-name [Methods]

NAME

get-name

DESCRIPTION

Geeft de naam van deze room terug.

SYNOPSIS

```
54      (define (get-name)
```

RETURN VALUE

```
string - de naam van deze room.
```

1.5.2.4. physical-room/set [Methods]

NAME

set

DESCRIPTION

Past de waarde van een element-type van deze physical-room aan.

SYNOPSIS

```
83      (define (set element-type value)
```

PARAMETERS

```
* element-type - de element-type waarvan de waarde aangepast moet worden.
* value - de nieuwe waarde van dit element-type.
```

RETURN VALUE

```
#<void>
```

1.5.3. physical/steward-server [Classes]

NAME

steward-server

DESCRIPTION

De server die op het steward toestel runt. Deze server leest instructions van de messenger, voert ze uit via zijn xbee-device en sendt het antwoord terug als een instruction-ret.

1.5.3.1. steward-server/new-steward-server [Constructors]

NAME

new-steward-server

DESCRIPTION

Maakt een nieuwe steward-server aan. Een steward-server bevindt zich in een bepaalde kamer, de naam van de kamer moet dezelfde zijn als het overeenkomstige steward object beheerst door de central-unit omdat de naam van de kamer gebruikt wordt om de communicatie port te bepalen.

SYNOPSIS

```
39 (define (new-steward-server room)
```

PARAMETERS

```
* room - de naam van de kamer waarin de steward-server zich bevindt
```

1.6. domotica/rules [Modules]

NAME

rule

DESCRIPTION

Het rule module bevat de klassen in verband met het rule-systeem. Rules laten toe om een planning te maken die automatisch instructies zendt aan de stewards. Het is bijvoordbeeld mogelijk om een rule te maken die elke maandag om 11.00 uur de temperatuur op 21°C zet.

1.6.1. rules/recurrence [Classes]

NAME

recurrence

DESCRIPTION

Deze klasse zorgt ervoor dat een bepaalde rule herhaald kan worden. De beschikbare herhalingen zijn :

- "once"
- "daily"
- "weekly"

1.6.1.1. recurrence/new-recurrence [Constructors]

NAME

new-recurrence

DESCRIPTION

Maakt een nieuw recurrence object aan. Het type en next-time bepalen het interval tussen een datum en zijn volgende herhaling.

De beschikare types zijn :

- "once"
- "daily"
- "weekly"

Er is ook een einddatum die het einde van herhalingen bepaald.

SYNOPSIS

```
50 (define (new-rec type next-time until)
```

PARAMETERS

```
* type - het type van de recurrence.
* next-time - het aantal seconden tussen een datum en zijn herhaling.
* until - het einddatum van de recurrence.
```

1.6.1.2. recurrence/get-end [Methods]

NAME

get-end

DESCRIPTION

Geeft het einddatum van deze recurrence terug.

SYNOPSIS

```
72   (define (get-end)
```

RETURN VALUE

```
date - het einddatum van deze recurrence.
```

1.6.1.3. recurrence/get-type [Methods]

NAME

get-type

DESCRIPTION

Geeft het type van deze recurrence terug.

SYNOPSIS

```
60   (define (get-type)
```

RETURN VALUE

```
string - het type van deze recurrence.
```

1.6.1.4. recurrence/next [Methods]

NAME

next

DESCRIPTION

Geeft de volgende herhaling van een datum terug.

SYNOPSIS

```
87   (define (next date)
```

PARAMETERS

```
* date - de datum waarvan de herhaling bekend wilt zijn.
```

RETURN VALUE

```
time-interval - een time-interval met de volgende herhaling en hetzelfde type recurrence.
#f - false wanneer het einddatum bereikt is.
```

1.6.2. rules/rule [Classes]

NAME

rule

DESCRIPTION

Deze klasse laat toe om automatisch element-types aan te passen op een specifiek tijdstip.

1.6.2.1. rule/new-rule [Constructors]

NAME

new-rule

DESCRIPTION

Maakt een nieuwe rule aan. Deze zet een element-type op een nieuwe waarde via execute. De rule kan alleen uitgevoerd worden wanneer de huidige datum deel is van zijn time-interval.

SYNOPSIS

```
29 (define (new-rule element-type value time-int)
```

PARAMETERS

```
* element-type - het element-type die aangepast gaat worden.
* value - de waarde waarop het element-type gezet moet worden.
* tim-int - het time-interval van deze rule.
```

1.6.2.2. rule/execute [Methods]

NAME

execute

DESCRIPTION

Probeert de rule uit te voeren. Deze wordt alleen uitgevoerd wanneer de huidige datum deelmaakt van het time-interval.

SYNOPSIS

```
101   (define (execute steward)
```

PARAMETERS

```
* steward - de steward waarop de rule uitgevoerd moet worden.
```

RETURN VALUE

```
#f - false wanneer de rule niet uitgevoerd werd.
#t - true bij het succesvol uitvoeren van de rule.
```

1.6.2.3. rule/get-element-type [Methods]

NAME

get-element-type

DESCRIPTION

Geeft het element-type van deze rule terug.

SYNOPSIS

```
39   (define (get-element-type)
```

RETURN VALUE

element-type - het element-type van de rule.

1.6.2.4. rule/get-interval [Methods]

NAME

get-interval

DESCRIPTION

Geeft het time-interval van deze rule terug.

SYNOPSIS

```
63   (define (get-interval)
```

RETURN VALUE

time-interval - het time-interval van de rule.

1.6.2.5. rule/get-value [Methods]

NAME

get-value

DESCRIPTION

Geeft de waarde van deze rule terug.

SYNOPSIS

```
51   (define (get-value)
```

RETURN VALUE

integer - de waarde van de rule.

1.6.2.6. rule/to-string [Methods]

NAME

to-string

DESCRIPTION

Zet de rule om naar een leesbare string.

SYNOPSIS

```
75   (define (to-string)
```

RETURN VALUE

string - een string-versie van de rule

1.6.3. rules/rule-manager [Classes]

NAME

rule-manager

DESCRIPTION

Een manager die rules beheerst. Elke steward heeft zijn eigen **rule-manager**.

1.6.3.1. rule-manager/new-rule-manager [Constructors]

NAME

new-rule-manager

DESCRIPTION

Maakt een nieuw rule-manager object aan.

SYNOPSIS

```
23 (define (new-rule-manager steward)
```

PARAMETERS

* steward - de steward die deze rule-manager beheerst.

1.6.3.2. rule-manager/add-rule [Methods]

NAME

add-rule

DESCRIPTION

Voeg een nieuwe rule toe aan deze manager.

SYNOPSIS

```
48     (define (add-rule rule)
```

PARAMETERS

* rule - de rule die toegevoegd moet worden.

RETURN VALUE

#<void>

1.6.3.3. rule-manager/execute [Methods]

NAME

execute

DESCRIPTION

Probeer alle rules van deze manager toe te passen op zijn steward.

SYNOPSIS

```
96      (define (execute)
```

RETURN VALUE

```
#<void>
```

1.6.3.4. rule-manager/get-rules [Methods]

NAME

get-rules

DESCRIPTION

Geeft een list met de rules die deze rule-manager beheerst terug.

SYNOPSIS

```
60      (define (get-rules)
```

RETURN VALUE

```
list - een list met de rules van deze manager.
```

1.6.3.5. rule-manager/get-steward [Methods]

NAME

get-steward

DESCRIPTION

Geeft de steward die deze manager beheerst terug.

SYNOPSIS

```
34      (define (get-steward)
```

RETURN VALUE

```
steward - de steward van deze manager.
```

1.6.3.6. rule-manager/remove-rule [Methods]

NAME

remove-rule

DESCRIPTION

Verwijder een rule van deze rule-manager.

SYNOPSIS

```
75      (define (remove-rule rule)
```

PARAMETERS

* rule - de rule die verwijdert moet worden.

RETURN VALUE

#f - false wanneer de rule niet gevonden werd.
#t - true wanneer de rule verwijderd werd.

1.6.4. rules/time-interval [Classes]

NAME

time-interval

DESCRIPTION

De **time-interval** klasse bepaalt of een datum deel uitmaakt van de datum (herhalingen inbegrepen) van deze interval.

1.6.4.1. time-interval/new-time-interval [Constructors]

NAME

new-time-interval

DESCRIPTION

Maakt een nieuwe time-interval object aan. Een time-interval begint op een datum en heeft een type recurrence.

SYNOPSIS

```
24 (define (new-time-interval date recurr)
```

PARAMETERS

date - de begin-datum van dit time-interval.
recurr - de recurrence van dit interval.

1.6.4.2. time-interval/get-date [Methods]

NAME

get-date

DESCRIPTION

Geeft de begindatum van het interval terug.

SYNOPSIS

```
34   (define (get-date)
```

RETURN VALUE

date - de begindatum van dit time-interval.

1.6.4.3. time-interval/get-recurrence [Methods]

NAME

get-recurrence

DESCRIPTION

Geeft de recurrence van dit time-interval terug.

SYNOPSIS

```
46   (define (get-recurrence)
```

RETURN VALUE

```
   recurrence - de recurrence van dit interval.
```

1.6.4.4. time-interval/is-on-time [Methods]

NAME

is-on-time

DESCRIPTION

Bepaalt of een datum deelmaakt van dit time-interval (als het gelijk is aan de begindatum of een van zijn herhalingen).

SYNOPSIS

```
62   (define (is-on-time time)
```

PARAMETERS

```
   * time - de datum die vergeleken moet worden.
```

RETURN VALUE

```
   #t - true als het deelmaakt van dit interval.
   #f - false als het niet deelmaakt van dit interval.
```

1.7. domotica/structure [Modules]

NAME

structure

DESCRIPTION

Het **structure** module bevat structuur klassen zoals map.

1.7.1. structure/get-ipv4-addrs [Methods]

NAME

get-ipv4-addrs

DESCRIPTION

Geeft het ipv4 adres van een machine.

Zie http://www.neilvandyke.org/racket-hostname/

1.7.2. structure/map [Classes]

NAME

map

DESCRIPTION

De **map** klasse is een klasse die verschillende tupels bevat. Elke tupel heeft een key (elke key in de **map** moet uniek zijn) en een element. Een element kan gevonden worden via zijn unieke key. Element is hier gezien als een algemene variabele en heeft geen verband met de element klasse van het internal module.

1.7.2.1. map/new-map [Constructors]

NAME

new-map

DESCRIPTION

Maakt een nieuwe lege map aan.

SYNOPSIS

```
23 (define (new-map)
```

1.7.2.2. map/add! [Methods]

NAME

add!

DESCRIPTION

Voeg een nieuwe tupel (key, element) toe aan de map. De key moet uniek zijn (er bestaat nog geen tupel met deze key).

SYNOPSIS

```
38      (define (add! key element)
```

PARAMETERS

```
* key - de key van de tupel.
* element - het element van de tupel.
```

RETURN VALUE

```
#<void>
```

1.7.2.3. map/get-elements [Methods]

NAME

get-elements

DESCRIPTION

geeft een list terug met alle elements die deze map bevat.

SYNOPSIS

```
92      (define (get-elements)
```

RETURN VALUE

```
list - een lijst met alle elements van deze map.
```

1.7.2.4. map/get-keys [Methods]

NAME

get-keys

DESCRIPTION

geeft een list terug met alle keys die deze map bevat.

SYNOPSIS

```
104     (define (get-keys)
```

RETURN VALUE

```
list - een lijst met alle keys van deze map.
```

1.7.2.5. map/key [Methods]

NAME

key

DESCRIPTION

Zoek naar een van de tupels die element bevat en geeft daarvan de **key** terug. Als er verschillende tupels zijn met hetzelfde element wordt maar een van de keys teruggegeven.

SYNOPSIS

```
120     (define (key element)
```

PARAMETERS

```
* element - het element van de gezochte tupel.
```

RETURN VALUE

```
any - de key van de gevonden tupel.
#f - false als geen tupel gevonden werd.
```

1.7.2.6. map/remove! [Methods]

NAME

remove!

DESCRIPTION

Verwijder een tupel van de map.

SYNOPSIS

```
53     (define (remove! key)
```

PARAMETERS

```
* key - de key van de tupel die verwijdert moet worden.
```

RETURN VALUE

```
#<void> - als de tupel verwijderd werd.
#f - false als er geen tupel met deze key was.
```

1.8. domotica/unit-test [Modules]

NAME

unit-test

DESCRIPTION

Het **unit-test** module bevat een reeks testen die de verschillende klassen van het domotica systeem testen. De klassen die getest

worden zijn :

- actuator
- device
- element
- hardware-device
- instruction
- messenger
- parser
- physical-room
- recurrence
- rule
- sensor
- steward
- time-interval
- xbee-simulation
- zigbee-instruction