SE 3XA3: Module Interface Specification
ScrumBot

Team 304, ScrumBot
Arkin Modi, modia1
Leon So, sol4
Timothy Choy, choyt2

Last Updated: March 12, 2020

MeetingTypes Module

Module

MeetingTypes

Uses

N/A

Syntax

Exported Constants

N/A

Exported Types

MeetingT = {Grooming, StandUp, Retrospective, SprintPlanning}

Exported Access Programs

None

Semantics

State Variables

None

State Invariant

None

Generic Dictionary Module

Generic Template Module

Dict(T)

Uses

N/A

Syntax

Exported Types

Dict = ?

Exported Constants

None

Exported Access Programs

Routine Name	In	Out	Exceptions
new Dict		Dict	
add	T		
remove	$\mathbb{N}$		KeyError
toSeq		set of ($\mathbb{N}$,T)	

Semantics

State Variables

D: set of ($\mathbb{N}$,T)
c: $\mathbb{N}$

State Invariant

$|D| \geq 0$
$c \geq 0$

Assumptions & Design Decision

- The Dict(T) constructor is called for each object instance before any other access routine is called for that object.

Access Routine Semantics

new Dict():

- transition: $D, c := \langle\rangle, 0$
- output: $out := self$

add(e):

- transition: $D := D \mid\mid \langle c,\ e\rangle, c := c + 1$

remove(id):

- transition: $D := D - \langle id, e\rangle$
- exception: $exc := \langle id, e\rangle \notin D \Rightarrow$ KeyError

toSeq():

- output: D

MeetingTypes Module

Module

MeetingTypes

Uses

N/A

Syntax

Exported Constants

N/A

Exported Types

MeetingT = {Grooming, StandUp, Retrospective, SprintPlanning}

Exported Access Programs

None

Semantics

State Variables

None

State Invariant

None

Generic Dictionary Module

Generic Template Module

Dict(T)

Uses

N/A

Syntax

Exported Types

Dict = ?

Exported Constants

None

Exported Access Programs

Routine Name	In	Out	Exceptions
new Dict(T)		Dict(T)	
add	T		
remove	$\mathbb{N}$		KeyError
to_seq		seq of ($\mathbb{N}$,T)	

Semantics

State Variables

d: seq of ($\mathbb{N}$,T)
c: $\mathbb{N}$

State Invariant

$|d| \geq 0$
$c \geq 0$

Assumptions & Design Decisions

- The Dict(T) constructor is called for each object instance before any other access routine is called for that object.

Access Routine Semantics

new Dict():

- transition: $d, c := \langle\rangle, 0$
- output: $out :=$ self

add(e):

- transition: $d := d \mid\mid \langle c,\ e\rangle, c := c + 1$

remove(id):

- transition: $d := d - \langle id, e\rangle$
- exception: $exc := \langle id, e\rangle \notin d \Rightarrow$ KeyError

to_seq():

- output: $out := d$ such that $(\forall i \in \mathbb{N} \mid 0 \leq i < |d| - 1 \cdot d[i] \leq d[i+1])$

MeetingList Module

Template Module

MeetingList is a Dict(Meeting)

Meeting Module

Module

Meeting

Uses

MeetingTypes

Syntax

Exported Constants

None

Exported Types

Meeting = ?

Exported Access Programs

Routine Name	In	Out	Exceptions
new Meeting	String, Date, Time, MeetingT	Meeting	
new Meeting	String, Date, Time, MeetingT, String	Meeting	
get_name		String	
get_date		Date	
get_time		Time	
get_type		MeetingT	
get_description		String	
set_description	String		

Semantics

State Variables

name: String
date: Date
time: Time

type: MeetingT
desc: String

State Invariant

None

Assumptions

- The Meeting constructor is called for each object instance before any other access routine is called for that object.

Access Routine Semantics

new Meeting(n, d, t, *type*)

- transition: $name,\ time,\ date,\ type,\ desc := n,\ d,\ t,\ type,$ None
- output: $out :=$ self

new Meeting(n, d, t, *type*, *desc*)

- transition: $name,\ time,\ date,\ type,\ desc := n,\ d,\ t,\ type,\ desc$
- output: $out :=$ self

get_name()

- output: $out := name$

get_date()

- output: $out := date$

get_time()

- output: $out := time$

get_type()

- output: $out := type$

get_description()

- output: $out := desc$

set_description(s)

- transition: $desc := s$

Task List Module

Template Module

TaskList is a Dict(Task)

Task Module

Module

Task

Uses

None

Syntax

Exported Constants

None

Exported Types

Task = ?

Exported Access Programs

Routine Name	In	Out	Exceptions
new Task	String, Date, Time	Task	
new Task	String, Date, Time, Details	Task	
get_deadline		(Date, Time)	
get_details		String	
get_feedback		seq of String	
add_feedback	String		
rm_feedback	$\mathbb{N}$		
set_details	String		

Semantics

State Variables

name: String
deadline: (Date, Time)
details: String
$feedback$: seq of String

State Invariant

None

Assumptions

- The Task constructor is called for each object instance before any other access routine is called for that object.

Access Routine Semantics

new Task(s, d, t)

- transition: $name, deadline, details := s, (d, t), \text{None}$
- output: $out :=$ self

new Task($s, d, t, details$)

- transition: $name, deadline, details := s, (d, t), details$
- output: $out :=$ self

get_deadline()

- output: $out := deadline$

get_details()

- output: $out := (details = \text{None} \Rightarrow \text{“No details”} \mid details)$

get_feedback()

- output: $out := feedback$

add_feedback(s)

- transition: $feedback := feedback \mid\mid s$

rm_feedback(s)

- transition: $feedback := feedback - s$

set_details(s)

- transition: $details := s$

Sprint Module

Module

Sprint

Uses

TaskList, Task

Syntax

Exported Constants

None

Exported Types

Sprint = ?

Exported Access Programs

Routine Name	In	Out	Exceptions
new Sprint	$\mathbb{N}$	Sprint	
get_tasks		seq of Task	
add_task	Task		
rm_task	$\mathbb{N}$		

Semantics

State Variables

$sprint\_num$: $\mathbb{N}$
$tasks$: TaskList

State Invariant

None

Assumptions

- The Sprint constructor is called for each object instance before any other access routine is called for that object.

Access Routine Semantics

new Sprint(n)

- transition: $sprint\_num, tasks := n, TaskList()$
- output: $out :=$ self

get_task(n)

- output: $out := tasks.to\_seq()$

add_task($task$)

- transition: $tasks := tasks.add(task)$

rm_task(n)

- transition: $tasks := tasks.renove(n)$

ProjectList Module

Template Module

ProjectList is a Dict(Project)

Project Module

Module

Project

Uses

Sprint, Tasklist, Task

Syntax

Exported Constants

None

Exported Types

Project = ?

Exported Access Programs

Routine Name	In	Out	Exceptions
new Project	String	Project	
new Project	String, String	Project	
get_desc		String	
get_meetings		seq of Meeting	
get_rqes		seq of String	
get_sprints		seq of Sprint	
set_desc	String		
add_meeting	Meeting		
add_rqe	String		
pop_sprint	$\mathbb{N}$		IndexError
push_sprint	Sprint		
rm_meeting	$\mathbb{N}$		
rm_rqe	$\mathbb{N}$		IndexError

Semantics

State Variables

name: String
desc: String
meetings: MeetingList
rqes: seq of String
sprints: seq of Sprint

State Invariant

None

Assumptions

- The Project constructor is called for each object instance before any other access routine is called for that object.

Access Routine Semantics

new Project(n)

- transition: $name, desc, rqes, sprints := n,\ \text{None},\ [],\ []$
- output: $out :=$ self

new Project(n, d)

- transition: $name, desc, rqes, sprints := n,\ d,\ [],\ []$
- output: $out :=$ self

get_desc()

- output: $out := (desc = \text{None} \Rightarrow \text{“No description”} \mid desc)$

get_meetings()

- output: $out := meetings.to_seq()$

get_rqes()

- output: $out := rqes$

get_sprints()

- output: $out := sprints$

set_desc(s)

- transition: $desc := s$

add_meeting($meeting$)

- transition: $meetings := meetings.add(meeting)$

add_rqe(s)

- transition: $rqes := rqes \mid\mid s$

pop_sprint()

- transition: $sprints := sprints[0 : |sprints| - 2]$
- exception: $exc := |sprints| = 0 \Rightarrow$ IndexError

push_sprint($sprint$)

- transition: $sprints := sprints \mid\mid sprint$

rm_meeting(n)

- transition: $meetings := meetings.remove(n)$

rm_rqe(n)

- transition: $rqes := rqes - rqes[n]$

Table 1: Revision History

Date	**Developer(s)**	**Change**
March 9, 2020	Timothy Choy	Create template, ScrumBot Module
March 10, 2020	Leon So	MeetingList module
March 11, 2020	Leon So	Meeting, MeetingTypes Modules
March 11, 2020	Leon So	Project, ProjectList, Meeting, MeetingTypes, Generic Dict Modules
March 11, 2020	Timothy Choy	Scrumbot Module
March 12, 2020	Leon So	Rename Meeting Module to Generic Meeting Module
March 12, 2020	Timothy Choy	Generic Meeting Module, Fixed Formatting
March 12, 2020	Arkin Modi	Fix Formatting
March 12, 2020	Leon So	Dict, Meeting, Project, Task, TaskList, Sprint Modules
March 12, 2020	Timothy Choy	Project, Meeting Modules