Disaster Event Module

Module

DisasterEvent

Uses

LatLng (android)

Syntax

Exported Types

DisasterEvent = ?

Exported Access Programs

Routine name	**In**	**Out**	**Exceptions**
DisasterEvent	String, LatLng, LatLng, $\mathbb{Z}$, $\mathbb{Z}$, $\mathbb{Z}$	DisasterEvent	
compareTo	DisasterEvent	$\mathbb{Z}$	
getType		String	
getLocation1		LatLng	
getLocation2		LatLng	
getYear		$\mathbb{Z}$	
getMonth		$\mathbb{Z}$	
getDay		$\mathbb{Z}$	

Semantics

State Variables

type: String
location1: LatLng
location2: LatLng
year: $\mathbb{Z}$
month: $\mathbb{Z}$
day: $\mathbb{Z}$

State Invariant

None

Assumptions

The constructor DisasterEvent is called for each object instance before any other access routine is called for that object. The constructor cannot be called on an existing object.

Access Routine Semantics

DisasterEvent(t, l_1, l_2, y, m, d):

- transition: $\mathit{type}, \mathit{location}1, \mathit{location}2, \mathit{year}, \mathit{month}, \mathit{day} := t, l_1, l_2, y, m, d$
- output: $\mathit{out} := \mathit{self}$

compareTo(other):

- output: $\mathit{out} := \mathit{year} < \mathit{other.year} \Rightarrow -1 | \mathit{year} > \mathit{other.year} \Rightarrow 1 | (\mathit{month} < \mathit{other.month} \Rightarrow -1 | \mathit{month} > \mathit{other.month} \Rightarrow 1 | (\mathit{day} < \mathit{other.day} \Rightarrow -1 | \mathit{day} > \mathit{other.day} \Rightarrow 1 | 0))$

getType():

- output: $\mathit{out} := \mathit{type}$

getLocation1():

- output: $\mathit{out} := \mathit{location}1$

getLocation2():

- output: $\mathit{out} := \mathit{location}2$

getYear():

- output: $\mathit{out} := \mathit{year}$

getMonth():

- output: $\mathit{out} := \mathit{month}$

getDay():

- output: $\mathit{out} := \mathit{day}$

Data Module

Module

Data

Uses

DisasterEvent

Syntax

Exported Types

None

Exported Access Programs

Routine name	**In**	**Out**	**Exceptions**
clear			
add	DisasterEvent		
getList	String	Sequence of DisasterEvent	
getTypes		Set of DisasterEvent	

Semantics

State Variables

data: HashMap of (String, Sequence of DisasterEvent)

State Invariant

None

Access Routine Semantics

clear():

- transition: $data := <>$

add(de):

- transition: $data.\text{getList}(de.\text{getType}) = data.\text{getList}(de.\text{getType}) \| < de >$

getList(key):

- output: $out := data.\text{get}(key)$

getTypes():

- output: $out := data.\text{keySet}$

New Parser Module

Module

NewParser

Uses

Data, DisasterEvent

Syntax

Exported Types

None

Exported Access Programs

Routine name	In	Out	Exceptions
firstParser	String		
secondParser	String		

Semantics

State Variables

None

State Invariant

None

Access Routine Semantics

firstParser(s):

- Opens a file f with name s. For each row, let r be an array of strings, representing the columns defined in f. Let $year, month, day, type, lat1, lng1, lat2, lng2 :=$ r[0].substring(0, 4), r[0].substring(4, 6), r[1], r[12], r[44], r[45],r[46], r[47]. Open a second file f' with name "c" $\|s$. For each row in f, write to f' the line: $year\|month\|day\|type\|lat1\|lng1\|lat2\|lng2$

secondParser(s):

- Used to parse files created from firstParser
- Opens a file f with name s. For each row, let r be an array of strings, representing the columns defined in f. Let $year, month, day, type, lat1, lng1, lat2, lng2 := r[0]$, r[1], r[2], r[3], r[4], r[5],r[6], r[7]. For each row in f,
  let $de :=$ DisasterEvent($year, month, day, type, lat1, lng1, lat2, lng2$). Data.add($de$)