Desease Simulator

1.1

Generated by Doxygen 1.8.20

Chapter 1

Namespace Index

1.1 Namespace List

Here is a list of all documented namespaces with brief descriptions:

Chapter 2

Hierarchical Index

2.1 Class Hierarchy

This inheritance list is sorted roughly, but not completely, alphabetically:

Chapter 3

Class Index

3.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

Chapter 4

Namespace Documentation

4.1 Actor Namespace Reference

Actor class, which is the base class for Disease objects.

Classes

- class Actor

4.1.1 Detailed Description

Actor class, which is the base class for Disease objects.

Author

Viviane Magalhães Siqueira (based on Professor Paulo Roma's work)

Date

10/08/2020

4.2 IDisease Namespace Reference

Disease Interface.

Classes

- class IDisease

  Interface IDisease allows setting the strength and growth condition of a disease.

Variables

- **ABC** = object

4.2.1 Detailed Description

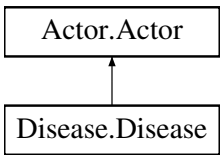
Disease Interface.

Chapter 5

Class Documentation

5.1 Actor.Actor Class Reference

Inheritance diagram for Actor.Actor:



Public Member Functions

- def __init__ (self)

  Construct a new Actor object.
- def getID (self)

  Used for testing.
- def Iteration (self)

  Used for testing.
- def act (self)

  Prints on screen in the format "Iteration <ID>: Actor <Actor ID>".
- def setLocation (self, x, y)

  Sets the cell coordinates of this object.
- def addedToWorld (self, world)

  Sets the world this actor is into.
- def getWorld (self)

  Gets the world this object in into.
- def getX (self)

  Gets the X coordinate of the cell this actor object is into.
- def getY (self)

  Gets the Y coordinate of the cell this actor object is into.
- def __str__ (self)

  Return a string with this actor ID and position.

5.1.1 Constructor & Destructor Documentation

5.1.1.1 __init__()

```
def Actor.Actor.__init__ (
              self )
```

Construct a new Actor object.

- Sets the initial values of its member variables.
- Sets the unique ID for the object and initializes the reference to the World object to which this Actor object belongs to null.
- The ID of the first Actor object is 0.
- The ID gets incremented by one each time a new Actor object is created.
- Sets the iteration counter to zero and initialize the location of the object to cell (0,0).

Reimplemented in Disease.Disease.

5.1.2 Member Function Documentation

5.1.2.1 act()

```
def Actor.Actor.act (
              self )
```

Prints on screen in the format "Iteration <ID>: Actor <Actor ID>".

The $< ID >$ is replaced by the current iteration number. $< ActorID >$ is replaced by the unique ID of the Actor object that performs the act(self) method.

For instance, the actor with ID 1 shows the following result on the output screen after its act(self) method has been called twice.

```
Iteration 0: Actor 1
Iteration 1: Actor 1
```

Reimplemented in Disease.Disease.

5.1.2.2 addedToWorld()

```
def Actor.Actor.addedToWorld (
              self,
              world )
```

Sets the world this actor is into.

Parameters

world	Reference to the World object this Actor object is added.

Exceptions

RuntimeError	when world is null.

5.1.2.3 getID()

```
def Actor.Actor.getID (
              self )
```

Used for testing.

Returns

ActorID

5.1.2.4 getWorld()

```
def Actor.Actor.getWorld (
              self )
```

Gets the world this object in into.

Returns

the world this object belongs to

5.1.2.5 getX()

```
def Actor.Actor.getX (
              self )
```

Gets the X coordinate of the cell this actor object is into.

Returns

the x coordinate of this Actor object.

5.1.2.6 getY()

```
def Actor.Actor.getY (
              self )
```

Gets the Y coordinate of the cell this actor object is into.

Returns

the y coordinate of this Actor object.

5.1.2.7 Iteration()

```
def Actor.Actor.Iteration (
              self )
```

Used for testing.

Returns

number of iterations

5.1.2.8 setLocation()

```
def Actor.Actor.setLocation (
              self,
              x,
              y )
```

Sets the cell coordinates of this object.

Parameters

x	the column.
y	the row.

Exceptions

ValueError	when x $<$ 0 or x $>$= world width,
ValueError	when y $<$ 0 or y $>$= world height,
RuntimeError	when the world is null.

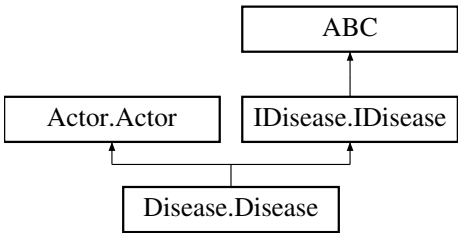
The documentation for this class was generated from the following file:

- C:/TI-Faculdade/PIG/AD2_Tkinter/Actor.py

5.2 Disease.Disease Class Reference

This Disease class is a sub-class of the Actor class.

Inheritance diagram for Disease.Disease:



Public Member Functions

- def __init__ (self)

  Constructor.
- def setGrowthCondition (self, lTemp, hTemp, gRate)

  Sets the disease growth rate, lower temperature and higher temperature.
- def getGrowthCondition (self)

  Returns the disease growth rate, lower temperature and higher temperature.
- def getQuadrant (self)

  Returns the quadrant of this disease.
- def getStrength (self)

  Return the disease strength of this object.
- def act (self)

  This method overrides the act () method in the Actor class .
- def __str__ (self)

  Return a string with the strength, growth and quadrant of this disease.

5.2.1 Detailed Description

This Disease class is a sub-class of the Actor class.

Author

Viviane Magalhães Siqueira (based on Professor Paulo Roma's work)

Date

10/08/2020

5.2.2 Constructor & Destructor Documentation

5.2.2.1 __init__()

```
def Disease.Disease.__init__ (
              self )
```

Constructor.

- Call its superclass's default constructor.
- Initialize the lower bound and the upper bound temperatures for the growth rate to 0.
- Set the growth rate to 0.
- Set the disease strength to 1.

Reimplemented from Actor.Actor.

5.2.3 Member Function Documentation

5.2.3.1 act()

```
def Disease.Disease.act (
              self )
```

This method overrides the act () method in the Actor class .

Check whether the object is in the region where the region temperature is within the lower bound and the upper bound temperatures for the object 's growth rate . If it is the case , multiply its strength with the growth rate .

Reimplemented from Actor.Actor.

5.2.3.2 getGrowthCondition()

```
def Disease.Disease.getGrowthCondition (
              self )
```

Returns the disease growth rate, lower temperature and higher temperature.

Returns

growth rate, lower temp and higher temp

5.2.3.3 getQuadrant()

```
def Disease.Disease.getQuadrant (
              self )
```

Returns the quadrant of this disease.

Returns

0, 1, 2 or 3.

5.2.3.4 getStrength()

```
def Disease.Disease.getStrength (
              self )
```

Return the disease strength of this object.

Returns

disease strength of the object.

Reimplemented from IDisease.IDisease.

5.2.3.5 setGrowthCondition()

```
def Disease.Disease.setGrowthCondition (
              self,
              lTemp,
              hTemp,
              gRate )
```

Sets the disease growth rate, lower temperature and higher temperature.

Parameters

lTemp	Lower bound temperature for the disease to grow at this gRate.
hTemp	Upper bound temperature for the disease to grow at this gRate.
gRate	The growth rate.

Reimplemented from IDisease.IDisease.

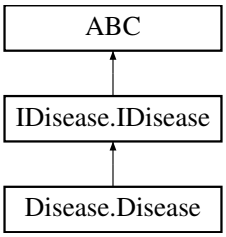
The documentation for this class was generated from the following file:

- C:/TI-Faculdade/PIG/AD2_Tkinter/Disease.py

5.3 IDisease.IDisease Class Reference

Interface IDisease allows setting the strength and growth condition of a disease.

Inheritance diagram for IDisease.IDisease:



Public Member Functions

- def setGrowthCondition (self, lTemp, hTemp, gRate)

  Set the growth condition of a Disease object to gRate.
- def getStrength (self)

  Return the disease strength of the object implements this interface.

5.3.1 Detailed Description

Interface IDisease allows setting the strength and growth condition of a disease.

Author

Paulo Cavalcanti

Date

22/02/2020

5.3.2 Member Function Documentation

5.3.2.1 setGrowthCondition()

```
def IDisease.IDisease.setGrowthCondition (
              self,
              lTemp,
              hTemp,
              gRate )
```

Set the growth condition of a Disease object to gRate.

The value of gRate gets multiplied to the current disease strength only when the disease is located in the world region with the average temperature in between the values of lTemp and hTemp.

Reimplemented in Disease.Disease.

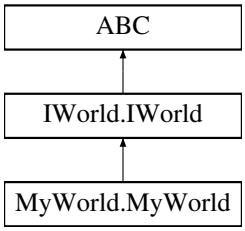
The documentation for this class was generated from the following file:

- C:/TI-Faculdade/PIG/AD2_Tkinter/IDisease.py

5.4 IWorld.IWorld Class Reference

Interface IWorld allows initializing and setting diseases for a world.

Inheritance diagram for IWorld.IWorld:



Public Member Functions

- def prepare (self)

  Prepare the world .
- def setTemp (self, quad, temp)

  Set the temperature of the region of the world to the value of temp .
- def getTemp (self, quad)

  Return the temperature of the world region with the ID of quadID .
- def initDiseases (self, numDisStr)

  Create Disease objects ; the number of the objects equals to the value passed in numDisStr .
- def getObjects (self)

  Return the list of objects in the class implementing this interface .
- def initTemps (self, tempStr)

  Set the temperature for each quadrant of the MyWorld according to the value of the tempStr .
- def getSumStrength (self)

  Return the total disease strength of all the diseases in the class implementing this interface .
- def initLocations (self, locationsStr, diseaseArr)

  Add each Disease object into the MyWorld object implementing this method according to the information in locationStr .
- def initGrowthConditions (self, growthStr, diseaseArr)

  Set the lower bound and upper bound temperature and the growth rate for each disease according to the input growthStr .

5.4.1 Detailed Description

Interface IWorld allows initializing and setting diseases for a world.

Author

Viviane Magalhães Siqueira (based on Professor Paulo Roma's work)

Date

10/08/2020

5.4.2 Member Function Documentation

5.4.2.1 getTemp()

```
def IWorld.IWorld.getTemp (
              self,
              quad )
```

Return the temperature of the world region with the ID of quadID .

The valid value is between zero and three inclusive .

Reimplemented in MyWorld.MyWorld.

5.4.2.2 initDiseases()

```
def IWorld.IWorld.initDiseases (
              self,
              numDisStr )
```

Create Disease objects ; the number of the objects equals to the value passed in numDisStr .

Return a list of object references to the created Disease objects .

An example of a valid numDisStr is below.

Ex: "2"

If numDisStr is None or it cannot be converted to a positive integer , print a message on screen " Check the NumDiseases line in simulation . config ." and return None .

No exceptions are thrown.

Reimplemented in MyWorld.MyWorld.

5.4.2.3 initGrowthConditions()

```
def IWorld.IWorld.initGrowthConditions (
              self,
              growthStr,
              diseaseArr )
```

Set the lower bound and upper bound temperature and the growth rate for each disease according to the input growthStr .

An example of a valid string for two Disease objects is:

Ex: "10.0 ,15.0 ,2.0;10.0 ,13.0 ,3.0"

If growthStr is empty or not in the correct format or does not have all the growth for all the Disease objects in the Disease array , print on screen " Check the DiseasesGrowth line in simulation . config ." and return -1.

Return 0 for a successful initialization of the Disease growth conditions . No exceptions are thrown .

Reimplemented in MyWorld.MyWorld.

5.4.2.4 initLocations()

```
def IWorld.IWorld.initLocations (
              self,
              locationsStr,
              diseaseArr )
```

Add each Disease object into the MyWorld object implementing this method according to the information in locationStr .

An example of a locationStr is "200 ,200;400 ,480". This means that the first Disease is planted at cell (200 ,200) and the second Disease is at cell (400 , 480).

If the locationStr is empty or not in the correct format or does not have all the cell coordinates of all the Disease objects , print on screen " Check the Locations line in simulation . config " and return -1.

Return 0 for a successful initialization of the Disease locations . No exceptions are thrown .

Reimplemented in MyWorld.MyWorld.

5.4.2.5 initTemps()

```
def IWorld.IWorld.initTemps (
              self,
              tempStr )
```

Set the temperature for each quadrant of the MyWorld according to the value of the tempStr .

An example of tempStr is below . The region temperatures for regions 0, 1, 2, and 3 are 12, 20, 50, and 100 , respectively .

Return 0 for a successful initialization of the quadrant temperatures . No exceptions are thrown . the quadrant temperatures . No exceptions are thrown .

Ex: "12;20;50;100"

If tempStr is empty or not in the correct format or does not have all the temperatures of all the regions , print on screen " Check the Temperature line in simulation . config ." and return -1.

Reimplemented in MyWorld.MyWorld.

5.4.2.6 prepare()

```
def IWorld.IWorld.prepare (
              self )
```

Prepare the world .

Open a text file named " simulation . config " in the current path (directly under the project directory). Parse the configuration file for the number of Disease objects , the cell locations of these objects , the growth rates , and the temperature ranges associated with individual growth rates . Read Section 4 on the content of the configuration file before reading the rest .

Reimplemented in MyWorld.MyWorld.

5.4.2.7 setTemp()

```
def IWorld.IWorld.setTemp (
              self,
              quad,
              temp )
```

Set the temperature of the region of the world to the value of temp .

The quadID indicates the region . The valid value is between [0, 3]. Any value of float is accepted for temp .

Reimplemented in MyWorld.MyWorld.

The documentation for this class was generated from the following file:

- C:/TI-Faculdade/PIG/AD2_Tkinter/IWorld.py

5.5 mapper.mapper Class Reference

Public Member Functions

- def __init__ (self, world, viewport, ydown=True, noDistortion=True)

  Constructor.
- def windowVecToViewport (self, x, y)

  Maps a single vector from world coordinates to viewport (screen) coordinates .
- def viewportToWindow (self, x, y)

  Maps a single point from screen coordinates to window (world) coordinates .
- def windowToViewport (self, *p)

  Maps points from world coordinates to viewport (screen) coordinates .

Public Attributes

- **world**
- **viewport**
- **fx**
- **fy**
- **ys**
- **f**
- **c_1**
- **c_2**

5.5.1 Constructor & Destructor Documentation

5.5.1.1 __init__()

```
def mapper.mapper.__init__ (
              self,
              world,
              viewport,
              ydown = True,
              noDistortion = True )
```

Constructor.

Parameters

world	window rectangle .
viewport	screen rectangle .
ydown	whether Y axis is upside down .
noDistortion	whether to use the same scale for both X and Y.

5.5.2 Member Function Documentation

5.5.2.1 viewportToWindow()

```
def mapper.mapper.viewportToWindow (
              self,
              x,
              y )
```

Maps a single point from screen coordinates to window (world) coordinates .

Parameters

x,y	given point .

Returns

a new point in world coordinates .

5.5.2.2 windowToViewport()

```
def mapper.mapper.windowToViewport (
              self,
             * p )
```

Maps points from world coordinates to viewport (screen) coordinates .

Parameters

p	a variable number of points .

Returns

two new points in screen coordinates .

5.5.2.3 windowVecToViewport()

```
def mapper.mapper.windowVecToViewport (
              self,
              x,
              y )
```

Maps a single vector from world coordinates to viewport (screen) coordinates .

Parameters

x,y	given vector .

Returns

a new vector in screen coordinates .

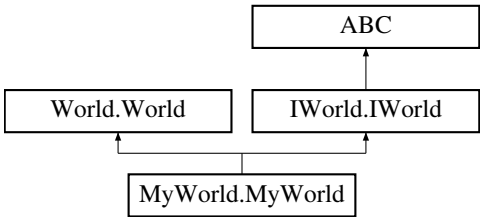
The documentation for this class was generated from the following file:

- C:/TI-Faculdade/PIG/AD2_Tkinter/mapper.py

5.6 MyWorld.MyWorld Class Reference

Call the constructor of the World class with the width and height of 720 and 640 cells , respectively .

Inheritance diagram for MyWorld.MyWorld:



Public Member Functions

- def __init__ (self, width=720, height=640)

  Constructor.
- def prepare (self)

  Prepare the world .
- def getObjects (self)

  Return the list of objects in the class implementing this interface .
- def initDiseases (self, numDisStr)

  Create Disease objects ; the number of the objects equals to the value passed in numDisStr .
- def initLocations (self, locationsStr, diseaseArr)

  Add each Disease object into the MyWorld object implementing this method according to the information in locationStr .
- def initGrowthConditions (self, growthStr, diseaseArr)

Set the lower bound and upper bound temperature and the growth rate for each disease according to the input growthStr .

- def initTemps (self, tempStr)

  Set the temperature for each quadrant of the MyWorld according to the value of the tempStr .

- def getSumStrength (self)

  Return the total disease strength of all the diseases in the class implementing this interface .

- def getTemp (self, quad)

  Return the temperature of the world region with the ID of quadID .

- def setTemp (self, quad, temp)

  Set the temperature of the region of the world to the value of temp .

- def act (self)

  This method overrides the act () method in the World class .

5.6.1 Detailed Description

Call the constructor of the World class with the width and height of 720 and 640 cells , respectively .

Initialize a list to keep the average temperature of each world region (quadrant).

Call the prepare () method .

5.6.2 Constructor & Destructor Documentation

5.6.2.1 __init__()

```
def MyWorld.MyWorld.__init__ (
              self,
              worldWidth = 720,
              worldHeight = 640 )
```

Constructor.

Creates a world with the given width and height.

- The maximum width and height are 1000.
- The maximum number of Actor objects in a cell is 5.

```
If worldWidth <= 0 or worldWidth > maximum width
    use the maximum width instead.
If worldHeight <=0 or worldHeight > maximum height
    use the maximum height instead.
```

Parameters

worldWidth	Width in number of cells
worldHeight	Height in number of cells

Reimplemented from World.World.

5.6.3 Member Function Documentation

5.6.3.1 act()

```
def MyWorld.MyWorld.act (
              self )
```

This method overrides the act () method in the World class .

This method prints :

" Iteration <ITRID >: World disease strength is <WorldDisease >" where <ITRID > is replaced by the current iteration number and <WorldDisease > is replaced by the returned value of getSumStrength () in 2 decimal places . An example is below .

Iteration 0: World disease strength is 2.00 Iteration 1: World disease strength is 3.00

Reimplemented from World.World.

5.6.3.2 getTemp()

```
def MyWorld.MyWorld.getTemp (
              self,
              quad )
```

Return the temperature of the world region with the ID of quadID .

The valid value is between zero and three inclusive .

Reimplemented from IWorld.IWorld.

5.6.3.3 initDiseases()

```
def MyWorld.MyWorld.initDiseases (
              self,
              numDisStr )
```

Create Disease objects ; the number of the objects equals to the value passed in numDisStr .

Return a list of object references to the created Disease objects .

An example of a valid numDisStr is below.

Ex: "2"

If numDisStr is None or it cannot be converted to a positive integer , print a message on screen " Check the NumDiseases line in simulation . config ." and return None .

No exceptions are thrown.

Reimplemented from IWorld.IWorld.

5.6.3.4 initGrowthConditions()

```
def MyWorld.MyWorld.initGrowthConditions (
              self,
              growthStr,
              diseaseArr )
```

Set the lower bound and upper bound temperature and the growth rate for each disease according to the input growthStr .

An example of a valid string for two Disease objects is:

Ex: "10.0 ,15.0 ,2.0;10.0 ,13.0 ,3.0"

If growthStr is empty or not in the correct format or does not have all the growth for all the Disease objects in the Disease array , print on screen "Check the DiseasesGrowth line in simulation.config." and return -1.

Return 0 for a successful

5.6.4 initialization of the Disease growth conditions .

5.6.5 No exceptions are thrown .

Reimplemented from IWorld.IWorld.

5.6.5.1 initLocations()

```
def MyWorld.MyWorld.initLocations (
              self,
              locationsStr,
              diseaseArr )
```

Add each Disease object into the MyWorld object implementing this method according to the information in locationStr .

An example of a locationStr is "200 ,200;400 ,480". This means that the first Disease is planted at cell (200 ,200) and the second Disease is at cell (400 , 480).

If the locationStr is empty or not in the correct format or does not have all the cell coordinates of all the Disease objects , print on screen " Check the Locations line in simulation . config " and return -1.

Return 0 for a successful initialization of the Disease locations . No exceptions are thrown .

Reimplemented from IWorld.IWorld.

5.6.5.2 initTemps()

```
def MyWorld.MyWorld.initTemps (
              self,
              tempStr )
```

Set the temperature for each quadrant of the MyWorld according to the value of the tempStr .

An example of tempStr is below . The region temperatures for regions 0, 1, 2, and 3 are 12, 20, 50, and 100 , respectively .

Reimplemented from IWorld.IWorld.

5.6.5.3 prepare()

```
def MyWorld.MyWorld.prepare (
              self )
```

Prepare the world .

Open a text file named " simulation . config " in the current path (directly under the project directory). Parse the configuration file for the number of Disease objects , the cell locations of these objects , the growth rates , and the temperature ranges associated with individual growth rates . Read Section 4 on the content of the configuration file before reading the rest .

Reimplemented from IWorld.IWorld.

5.6.5.4 setTemp()

```
def MyWorld.MyWorld.setTemp (
              self,
              quad,
              temp )
```

Set the temperature of the region of the world to the value of temp .

The quadID indicates the region . The valid value is between [0, 3]. Any value of float is accepted for temp .

Reimplemented from IWorld.IWorld.

The documentation for this class was generated from the following file:

- C:/TI-Faculdade/PIG/AD2_Tkinter/MyWorld.py

5.7 SimulationPanel.SimulationPanel Class Reference

Public Member Functions

- def __init__ (self, world, canvas)

  Constructor.
- def fromCoordinates (self, location)

  Create a dictionary with all the coordinates and strength of diseases.
- def draw (self)

  Draws the elements on the canvas.
- def resize (self, e)

  resizes the window and elements
- def printData (self, e)

  Save all diseases in the current world in a file, in the same format as simulation.config .
- def mousePressed (self, e)

  Press the mouse button 1 to indicate the center of new circles to be added.

Public Attributes

- **world**
- **canvas**
- **wvmap**
- **width**
- **height**
- **diseases**
- **dict**
- **raio**

5.7.1 Constructor & Destructor Documentation

5.7.1.1 __init__()

```
def SimulationPanel.SimulationPanel.__init__ (
            self,
            world,
            canvas )
```

Constructor.

Parameters

world	.
canvas	.

5.7.2 Member Function Documentation

5.7.2.1 fromCoordinates()

```
def SimulationPanel.SimulationPanel.fromCoordinates (
              self,
              location )
```

Create a dictionary with all the coordinates and strength of diseases.

Parameters

location	coordinates list.

5.7.2.2 mousePressed()

```
def SimulationPanel.SimulationPanel.mousePressed (
              self,
              e )
```

Press the mouse button 1 to indicate the center of new circles to be added.

Growth parameters can have any fixed ratio and temperatures minimum and maximum can be one to memos and one more, respectively, in relation to temperature of the quadrant where the point was inserted.

Parameters

e	(event) .

5.7.2.3 printData()

```
def SimulationPanel.SimulationPanel.printData (
              self,
              e )
```

Save all diseases in the current world in a file, in the same format as simulation.config .

Parameters

e	(event).

5.7.2.4 resize()

```
def SimulationPanel.SimulationPanel.resize (
              self,
              e )
```

resizes the window and elements

Parameters

e	(event).

The documentation for this class was generated from the following file:

- C:/TI-Faculdade/PIG/AD2_Tkinter/SimulationPanel.py

5.8 Timer.Timer Class Reference

Public Member Functions

- def **__init__** (self, root, callback, delay)
- def run (self)
- def stop (self)
- def restart (self)

Public Attributes

- **root**
- **callback**
- **delay**
- **task**

5.8.1 Detailed Description

```
Keep packing ( drawing ) circles , after a certain time interval .
```

5.8.2 Member Function Documentation

5.8.2.1 restart()

```
def Timer.Timer.restart (
              self )
```

```
Restart the drawing process .
```

5.8.2.2 run()

```
def Timer.Timer.run (
              self )

" Run the callback function every delay ms .
```

5.8.2.3 stop()

```
def Timer.Timer.stop (
              self )

 Stop the drawing process .
```

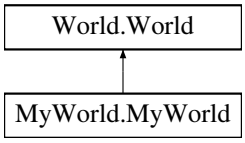
The documentation for this class was generated from the following file:

- C:/TI-Faculdade/PIG/AD2_Tkinter/Timer.py

5.9 World.World Class Reference

Class for holding Actor objects in cells of a grid in the world.

Inheritance diagram for World.World:



Public Member Functions

- def __init__ (self, worldWidth, worldHeight)
  Constructor.
- def createGrid (self, h, w, d)
  Initializes each object of the array as None.
- def __str__ (self)
  Return a string representation of the grid.
- def __repr__ (self)
  Return a string representation of the grid.
- def act (self)
  Blank method body.
- def addObject (self, objct, x, y)
  Adds a new actor to this world at a given position.

- def getHeight (self)

  Returns the world height.
- def getWidth (self)

  Returns the world width.
- def getDepth (self)

  Returns the world depth.
- def numberOfObjects (self)

  Returns the total number of objects in this world.
- def getObjects (self)

  Returns an array with all Actor objects in this world.
- def setGrid (self, aGrid, numObjs)

  It checks if aGrid is a 3D array with the same positive length in each dimension.

5.9.1 Detailed Description

Class for holding Actor objects in cells of a grid in the world.

The world is represented by a 2 dimensional array of cells, with the specified width and height. One cell can keep at most 5 Actor objects.

Author

Viviane Magalhães Siqueira (based on Professor Paulo Roma's work)

Date

10/08/2020

5.9.2 Constructor & Destructor Documentation

5.9.2.1 __init__()

```
def World.World.__init__ (
              self,
              worldWidth,
              worldHeight )
```

Constructor.

Creates a world with the given width and height.

- The maximum width and height are 1000.
- The maximum number of Actor objects in a cell is 5.

```
If worldWidth <= 0 or worldWidth > maximum width
    use the maximum width instead.
If worldHeight <=0 or worldHeight > maximum height
    use the maximum height instead.
```

Parameters

worldWidth	Width in number of cells
worldHeight	Height in number of cells

Reimplemented in MyWorld.MyWorld.

5.9.3 Member Function Documentation

5.9.3.1 __repr__()

```
def World.World.__repr__ (
            self )
```

Return a string representation of the grid.

List by depth. Each slice is height x width.

Returns

string with the grid.

See also

https://www.ict.social/python/basics/multidimensional-lists-in-python

5.9.3.2 __str__()

```
def World.World.__str__ (
            self )
```

Return a string representation of the grid.

List by width. Each slice is height x depth. (List by height. Each slice is width x depth)

Returns

string with the grid.

5.9.3.3 act()

```
def World.World.act (
              self )
```

Blank method body.

Overriden in subclasses as appropriate

Reimplemented in MyWorld.MyWorld.

5.9.3.4 addObject()

```
def World.World.addObject (
              self,
              objct,
              x,
              y )
```

Adds a new actor to this world at a given position.

- The new object will be added at the cell (x,y) if there are less than 5 objects in this cell.
- Be sure to make the added object know that it is in this world and it is at this cell.
- Check which methods of the Actor class to call.

Parameters

object	the object to be added at this cell (x, y)
x	the column
y	the row

Returns

number of objects in cell (x,y).

Exceptions

SyntaxError	when already max number of objects are in that cell
ValueError	if x or y is not in the valid range
NameError	if the object is null

5.9.3.5 createGrid()

```
def World.World.createGrid (
```

```
                self,
                h,
                w,
                d )
```

Initializes each object of the array as None.

Parameters

h	grid height.
w	grid width.
d	grid depth.

Returns

grid.

5.9.3.6 getDepth()

```
def World.World.getDepth (
                self )
```

Returns the world depth.

Returns

the world depth.

5.9.3.7 getHeight()

```
def World.World.getHeight (
                self )
```

Returns the world height.

Returns

the world height.

5.9.3.8 getObjects()

```
def World.World.getObjects (
              self )
```

Returns an array with all Actor objects in this world.

Returns

Array of Actor objects that are in this world.

Comments:

- Each class in Java is a subclass of the Object class.
- Observe that you use the implicit upcast where you assign an Actor object (sub-class) in an element of the Object array.

Reimplemented in MyWorld.MyWorld.

5.9.3.9 getWidth()

```
def World.World.getWidth (
              self )
```

Returns the world width.

Returns

the world width.

5.9.3.10 numberOfObjects()

```
def World.World.numberOfObjects (
              self )
```

Returns the total number of objects in this world.

Returns

Total number of objects in this world.

5.9.3.11 setGrid()

```
def World.World.setGrid (
              self,
              aGrid,
              numObjs )
```

It checks if aGrid is a 3D array with the same positive length in each dimension.

If so, it sets the grid to aGrid and the other private fields of class World to the dimension lengths of aGrid and numObjs.

Note that some checks are omitted. For example, no check is performed to make sure that numObjs is consistent with the number of Actor objects in aGrid.

Each Actor object in aGrid has to be set to this World object.

Parameters

aGrid	reference to a 3D array of Actor objects.
numObjs	the number of Actor objects in aGrid.

Exceptions

ValueError	if the length of each dimension is out of range or 2nd/3rd dimension has different lengths.

The documentation for this class was generated from the following file:

- C:/TI-Faculdade/PIG/AD2_Tkinter/World.py

Index