RSim

Generated by Doxygen 1.12.0

Chapter 1

Hierarchical Index

1.1 Class Hierarchy

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

Chapter 2

Class Index

2.1 Class List

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

Chapter 3

File Index

3.1 File List

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

Chapter 4

Class Documentation

4.1 config._Config_Visual Class Reference

Public Member Functions

- __init__ (self, config)

  Initializes the _Config_Visual class from a config parser object.
- str __repr__ (self)

  Returns a string representation of the visual configuration.

Public Attributes

- **window_name** = config.get('VISUAL', 'window_name', fallback=DEFAULT_WINDOW_NAME)
- **window_width** = config.getint('VISUAL', 'window_width', fallback=DEFAULT_WINDOW_WIDTH)
- **window_height** = config.getint('VISUAL', 'window_height', fallback=DEFAULT_WINDOW_HEIGHT)

4.1.1 Constructor & Destructor Documentation

4.1.1.1 __init__()

```
config._Config_Visual.__init__ (
              self,
              config)
```

Initializes the _Config_Visual class from a config parser object.

This constructor retrieves visual configuration parameters such as window name, width, and height from the provided config object.

Parameters

config	A configparser.ConfigParser object containing the configuration.

4.1.2 Member Function Documentation

4.1.2.1 __repr__()

```
str config._Config_Visual.__repr__ (
              self)
```

Returns a string representation of the visual configuration.

Returns

A formatted string describing the visual configuration settings.

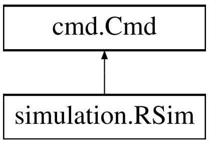
The documentation for this class was generated from the following file:

- config.py

4.2 cmd.Cmd Class Reference

Class for handling terminal commands and interaction in the simulation.

Inheritance diagram for cmd.Cmd:



Public Member Functions

- startCmd (cls)
- stopCmd (cls)
- processCmd (cls, command)

  Processes commands entered in the terminal.

Public Attributes

- **input** = threading.Thread(target=cls._handle_input)
- int **save_duration** = 60

Static Public Attributes

- **lock** = threading.Lock()

  Lock for synchronizing access to terminal input.

Protected Member Functions

- _handle_input (cls)

  Handles terminal input for commands.
- _save (cls)

  Saves the current state of the simulation.
- _clear (cls)

  Clears the terminal screen and resets the input prompt.
- _spawn (cls, args)

  Spawns entities or food in the simulation.
- _help (cls)

  Displays the help message with available commands.

Protected Attributes

- bool **_running** = "help":

4.2.1 Detailed Description

Class for handling terminal commands and interaction in the simulation.

4.2.2 Member Function Documentation

4.2.2.1 _clear()

```
cmd.Cmd._clear (
              cls)   [protected]
```

Clears the terminal screen and resets the input prompt.

This method determines the appropriate clear command based on the operating system.

4.2.2.2 _handle_input()

```
cmd.Cmd._handle_input (
              cls)   [protected]
```

Handles terminal input for commands.

This method continuously reads input from the terminal in a separate thread.

Note

> The input thread will stop when _running is set to False.

4.2.2.3 _help()

```
cmd.Cmd._help (
              cls)  [protected]
```

Displays the help message with available commands.

This method prints out a list of all commands that the user can enter.

4.2.2.4 _save()

```
cmd.Cmd._save (
              cls)  [protected]
```

Saves the current state of the simulation.

This method attempts to save the current simulation state and handles any errors.

4.2.2.5 _spawn()

```
cmd.Cmd._spawn (
              cls,
              args)  [protected]
```

Spawns entities or food in the simulation.

Parameters

args	The arguments for the spawn command.

This method interprets the spawn command and creates the specified element type.

4.2.2.6 processCmd()

```
cmd.Cmd.processCmd (
              cls,
              command)
```

Processes commands entered in the terminal.

Parameters

command	The command string entered by the user.

This method interprets the command and executes the corresponding action.

4.2.2.7 startCmd()

```
cmd.Cmd.startCmd (
              cls)
```

```
Starts the command input thread.
```

4.2.2.8 stopCmd()

```
cmd.Cmd.stopCmd (
              cls)
```

```
Waits for the command input thread to finish.
```

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

- cmd.py

4.3 config.Config Class Reference

Public Member Functions

- init (self, config_file=path.PATH_CONFIG)

    Initializes the configuration from a specified file.
- default (self, config_file=path.PATH_CONFIG)

    Creates and writes a default configuration file.
- str __repr__ (self)

    Returns a string representation of the Config class.

Static Public Attributes

- visual = None

    A class for managing the configuration settings of the application.

4.3.1 Member Function Documentation

4.3.1.1 __repr__()

```
str config.Config.__repr__ (
              self)
```

Returns a string representation of the Config class.

Returns

A formatted string describing the current configuration settings.

4.3.1.2 default()

```
config.Config.default (
              self,
              config_file = path.PATH_CONFIG)
```

Creates and writes a default configuration file.

This method initializes default visual settings and writes them to the specified config file.

Parameters

config_file	The path to the configuration file (default: path.PATH_CONFIG).

4.3.1.3 init()

```
config.Config.init (
              self,
              config_file = path.PATH_CONFIG)
```

Initializes the configuration from a specified file.

This class method checks if the specified config file exists. If not, it creates a default config file. It then reads the configuration and initializes the visual settings.

Parameters

config_file	The path to the configuration file (default: path.PATH_CONFIG).

4.3.2 Member Data Documentation

4.3.2.1 visual

```
config.Config.visual = None  [static]
```

A class for managing the configuration settings of the application.

This class provides methods to initialize configuration from a file, create default configurations, and access visual settings.

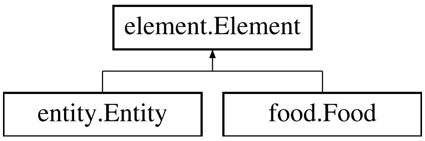
The documentation for this class was generated from the following file:

- config.py

4.4 element.Element Class Reference

Class representing an element on the map.

Inheritance diagram for element.Element:



Public Member Functions

- __init__ (self, coord)

  Initializes an element with a position.
- str __repr__ (self)

  Represents the Element as a string.
- __len__ (cls)

  Returns the number of elements.
- x (self)

  Property getter for x-coordinate.
- x (self, value)

  Property setter for x-coordinate.
- y (self)

  Property getter for y-coordinate.
- y (self, value)

  Property setter for y-coordinate.
- coord (self)

  Property getter for coordinates.
- coord (self, value)

  Property setter for coordinates.
- new (cls, coord)

  Creates and adds a new element to the list.
- delete (cls, elem)

  Removes an element from the list.
- int len (cls)

  Returns the number of elements.
- float distance (self, elem)

  Calculates the Euclidean distance from the element to another element.

Public Attributes

- **x** = coord[0]
- **y** = coord[1]

Static Public Attributes

- list **list** = [ ]

  List of all elements.

Protected Attributes

- _**x** = value
- _**y** = value

4.4.1 Detailed Description

Class representing an element on the map.

4.4.2 Constructor & Destructor Documentation

4.4.2.1 __init__()

```
element.Element.__init__ (
              self,
              coord)
```

Initializes an element with a position.

Parameters

coord	A tuple (x, y) representing the coordinates of the element.

Exceptions

TypeError	if 'coord' is not a tuple or its elements are not integers.
ValueError	if 'coord' does not have a length of 2.
ValueError	if x or y are out of bounds.

Reimplemented in entity.Entity, and food.Food.

4.4.3 Member Function Documentation

4.4.3.1 __len__()

```
element.Element.__len__ (
              cls)
```

Returns the number of elements.

Returns

The length of the element list.

4.4.3.2 __repr__()

```
str element.Element.__repr__ (
              self)
```

Represents the Element as a string.

Returns

A string representation of the element's coordinates.

Reimplemented in entity.Entity, and food.Food.

4.4.3.3 coord() [1/2]

```
element.Element.coord (
              self)
```

Property getter for coordinates.

Returns

A tuple (x, y) representing the element's coordinates.

4.4.3.4 coord() [2/2]

```
element.Element.coord (
              self,
              value)
```

Property setter for coordinates.

Parameters

value	A tuple (x, y) representing the new coordinates.

Exceptions

TypeError	if 'value' is not a tuple.
ValueError	if 'value' does not have a length of 2.

```
Set the coordinates of the element with validation.
```

4.4.3.5 delete()

```
element.Element.delete (
              cls,
              elem)
```

Removes an element from the list.

Parameters

elem	The element to remove from the list.

4.4.3.6 distance()

```
float element.Element.distance (
              self,
              elem)
```

Calculates the Euclidean distance from the element to another element.

Parameters

elem	An instance of Element representing the position to which the distance is calculated.

Returns

The Euclidean distance between the element's position and the given position.

4.4.3.7 len()

```
int element.Element.len (
              cls)
```

Returns the number of elements.

Returns

The length of the element list.

4.4.3.8 new()

```
element.Element.new (
              cls,
              coord)
```

Creates and adds a new element to the list.

Parameters

coord	A tuple (x, y) representing the coordinates of the new element.

Reimplemented in entity.Entity, and food.Food.

4.4.3.9 x() [1/2]

```
element.Element.x (
              self)
```

Property getter for x-coordinate.

Returns

The x-coordinate of the element.

4.4.3.10 x() [2/2]

```
element.Element.x (
              self,
              value)
```

Property setter for x-coordinate.

Parameters

value	New x-coordinate.

Exceptions

TypeError	if the x-coordinate is not an integer.
ValueError	if the x-coordinate is out of bounds.

4.4.3.11 y() [1/2]

```
element.Element.y (
              self)
```

Property getter for y-coordinate.

Returns

The y-coordinate of the element.

4.4.3.12 y() [2/2]

```
element.Element.y (
              self,
              value)
```

Property setter for y-coordinate.

Parameters

value	New y-coordinate.

Exceptions

TypeError	if the y-coordinate is not an integer.
ValueError	if the y-coordinate is out of bounds.

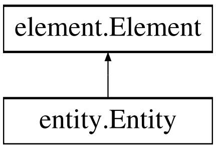
The documentation for this class was generated from the following file:

- element.py

4.5 entity.Entity Class Reference

Class representing an entity that extends *Element* and includes properties for energy, time, and age.

Inheritance diagram for entity.Entity:



Public Member Functions

- __init__ (self, coord, energy, time=0)

  Initializes an entity with a position, energy, and time.
- str __repr__ (self)

  Returns a string representation of the entity, including its position, energy, time, and age.
- energy (self)

  Property getter for the entity's energy.
- energy (self, value)

  Property setter for energy.
- time (self)

  Property getter for time.
- time (self, value)

  Property setter for time.
- age (self)

  Property getter for age.
- new (cls, coord, energy, time=0)

  Class method to create a new entity and add it to the class list.
- generate (cls)

  Class method to generate an entity at a random position.
- eat (self)

  Allows the entity to eat food located at the same coordinates.
- possibleMoves (self)

  Determines possible moves based on the entity's position and the map size.
- tuple rdmMove (self)

  Randomly move the entity to a neighboring position.
- move (self)

  Move the entity by choosing a random valid direction or towards food.
- moveTowardsFood (self)

  Finds and moves towards the closest food within the entity's viewing range.
- **reproduction** (self)
- bool **survive** (self)

Public Member Functions inherited from element.Element

- __len__ (cls)

  Returns the number of elements.
- x (self)

  Property getter for x-coordinate.
- x (self, value)

  Property setter for x-coordinate.
- y (self)

  Property getter for y-coordinate.
- y (self, value)

  Property setter for y-coordinate.
- coord (self)

  Property getter for coordinates.
- coord (self, value)

  Property setter for coordinates.
- delete (cls, elem)

  Removes an element from the list.
- int len (cls)

  Returns the number of elements.
- float distance (self, elem)

  Calculates the Euclidean distance from the element to another element.

Public Attributes

- **energy** = energy
- **time** = time
- int **range** = Entity.RANGE_DEF
- int **id** = Entity.nEntities
- int **energy** = 1
- **x**
- **y**

Public Attributes inherited from element.Element

- **x** = coord[0]
- **y** = coord[1]

Static Public Attributes

- int **NBYTES_ID** = 2
- int **NBYTES_ENERGY** = 2

  Number of bytes for energy in the binary representation.
- int **NBYTES_TIME** = 2

  Number of bytes for time in the binary representation.
- int **ENERGY_MAX** = 9999

  Maximum possible energy value for an entity.
- int **ENERGY_DEF** = 200

  Default energy value when an entity is generated.
- int **TIME_MAX** = 4000

Maximum possible time value for an entity.
- int **RANGE_DEF** = 10

  Default viewing range for the entity.
- list **list** = [ ]

  Class-level list to store all instances of Entity.
- int **nEntities** = 0
- int **MAX_ID** = 2 ∗∗ (NBYTES_ID ∗ 8)
- int **TIME_IN_AGE** = 40
- **AGE_REPROD** = range(18, 60)
- int **MIN_REPROD** = 500
- int **REPROD** = 200

Static Public Attributes inherited from element.Element

- list **list** = [ ]

  List of all elements.

Protected Attributes

- int **_energy** = Entity.ENERGY_MAX
- int **_time** = -1
- int **_age** = 0

Protected Attributes inherited from element.Element

- **_x** = value
- **_y** = value

Static Protected Attributes

- **_ln_10** = log(10)

4.5.1 Detailed Description

Class representing an entity that extends *Element* and includes properties for energy, time, and age.

4.5.2 Constructor & Destructor Documentation

4.5.2.1 __init__()

```
entity.Entity.__init__ (
              self,
              coord,
              energy,
              time = 0)
```

Initializes an entity with a position, energy, and time.

Parameters

coord	A tuple (x, y) representing the coordinates of the entity.
energy	An integer representing the initial energy of the entity.
time	An integer representing the time the entity has lived (default is 0).

Reimplemented from element.Element.

4.5.3 Member Function Documentation

4.5.3.1 __repr__()

```
str entity.Entity.__repr__ (
              self)
```

Returns a string representation of the entity, including its position, energy, time, and age.

Returns

A string in the format "pos=(x, y), energy=..., time=..., age=...".

Reimplemented from element.Element.

4.5.3.2 age()

```
entity.Entity.age (
              self)
```

Property getter for age.

Returns

The age of the entity, calculated from the time.

4.5.3.3 eat()

```
entity.Entity.eat (
              self)
```

Allows the entity to eat food located at the same coordinates.

This method checks if the entity's energy is less than the maximum energy. If food is found at the same location, the entity consumes it, increasing its energy.

4.5.3.4 energy() [1/2]

```
entity.Entity.energy (
              self)
```

Property getter for the entity's energy.

Returns

The current energy of the entity.

4.5.3.5 energy() [2/2]

```
entity.Entity.energy (
              self,
              value)
```

Property setter for energy.

Parameters

value	New energy value for the entity.

Exceptions

TypeError	if energy is not an integer.

Clamps the energy to [0, ENERGY_MAX] if the input exceeds those limits.

4.5.3.6 generate()

```
entity.Entity.generate (
              cls)
```

Class method to generate an entity at a random position.

Uses the Map class to generate a random coordinate and creates a new entity with default energy.

4.5.3.7 move()

```
entity.Entity.move (
              self)
```

Move the entity by choosing a random valid direction or towards food.

The entity will first try to eat if food is available, then move towards food if it is nearby. If no food is found, it makes a random move.

4.5.3.8 moveTowardsFood()

```
entity.Entity.moveTowardsFood (
              self)
```

Finds and moves towards the closest food within the entity's viewing range.

Searches the `Food.list` for the closest food that is within `self.range` of the entity. If no food is found, the entity does not move.

Returns

A tuple of (dx, dy) representing the movement direction, or None if no food is found.

4.5.3.9 new()

```
entity.Entity.new (
              cls,
              coord,
              energy,
              time = 0)
```

Class method to create a new entity and add it to the class list.

Parameters

coord	A tuple (x, y) representing the coordinates of the new entity.
energy	Initial energy value of the new entity.
time	Initial time value of the new entity (default is 0).

Returns

The total number of entities after adding the new one.

Reimplemented from element.Element.

4.5.3.10 possibleMoves()

```
entity.Entity.possibleMoves (
              self)
```

Determines possible moves based on the entity's position and the map size.

Returns

A list of valid neighboring coordinates.

4.5.3.11 rdmMove()

```
tuple entity.Entity.rdmMove (
              self)
```

Randomly move the entity to a neighboring position.

Returns

A tuple of (dx, dy) representing the movement direction.

4.5.3.12 time() [1/2]

```
entity.Entity.time (
              self)
```

Property getter for time.

Returns

The current time value of the entity.

4.5.3.13 time() [2/2]

```
entity.Entity.time (
              self,
              value)
```

Property setter for time.

Parameters

value	New time value for the entity.

Exceptions

TypeError	if time is not an integer.

Sets time; if time is invalid, resets it to -1 and sets energy and age to default values.

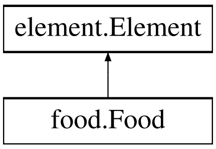
The documentation for this class was generated from the following file:

- entity.py

4.6 food.Food Class Reference

Class representing food objects in the simulation.

Inheritance diagram for food.Food:



Public Member Functions

- __init__ (self, coord, pts=PTS_DEFAULT)

  Initializes a new food object with coordinates and point value.
- str __repr__ (self)

  Represents the Food object as a string.
- pts (self)

  Property getter for the point value.
- pts (self, value)

  Property setter for the point value.
- new (cls, coord, pts)

  Creates a new food object and adds it to the list.
- generate (cls)

  Generates a new food object at a random map coordinate.

Public Member Functions inherited from element.Element

- __len__ (cls)

  Returns the number of elements.
- x (self)

  Property getter for x-coordinate.
- x (self, value)

  Property setter for x-coordinate.
- y (self)

  Property getter for y-coordinate.
- y (self, value)

  Property setter for y-coordinate.
- coord (self)

  Property getter for coordinates.
- coord (self, value)

  Property setter for coordinates.
- delete (cls, elem)

  Removes an element from the list.
- int len (cls)

  Returns the number of elements.
- float distance (self, elem)

  Calculates the Euclidean distance from the element to another element.

Public Attributes

- **pts** = pts

Public Attributes inherited from element.Element

- **x** = coord[0]
- **y** = coord[1]

Static Public Attributes

- int **NBYTES_PTS** = 2

  Number of bytes allocated for food points.
- int **PTS_MAX** = 2∗∗(NBYTES_PTS∗8)

  Maximum point value a food item can have.
- int **PTS_DEFAULT** = 100

  Default point value for new food items.
- list **list** = [ ]

  Class-level list to store all food items.
- int **MAXFOODS_DEF** = 50
- int **maxFoods** = MAXFOODS_DEF

Static Public Attributes inherited from element.Element

- list **list** = [ ]

  List of all elements.

Protected Attributes

- int **_pts** = Food.PTS_DEFAULT

Protected Attributes inherited from element.Element

- **_x** = value
- **_y** = value

4.6.1 Detailed Description

Class representing food objects in the simulation.

This class inherits from Element and represents food items placed on the map. Each food object has a position and a point value (`pts`), which is the energy it provides when consumed.

4.6.2 Constructor & Destructor Documentation

4.6.2.1 __init__()

```
food.Food.__init__ (
                self,
                coord,
                pts = PTS_DEFAULT)
```

Initializes a new food object with coordinates and point value.

Parameters

coord	A tuple (x, y) representing the coordinates of the food item on the map.
pts	An integer representing the energy value of the food, default is 100.

Reimplemented from element.Element.

4.6.3 Member Function Documentation

4.6.3.1 __repr__()

```
str food.Food.__repr__ (
              self)
```

Represents the Food object as a string.

Returns

A string representation of the food's coordinates and points.

Reimplemented from element.Element.

4.6.3.2 generate()

```
food.Food.generate (
              cls)
```

Generates a new food object at a random map coordinate.

If a valid coordinate is found, a new food item is created and added to the list.

4.6.3.3 new()

```
food.Food.new (
              cls,
              coord,
              pts)
```

Creates a new food object and adds it to the list.

Parameters

coord	A tuple (x, y) representing the coordinates of the food.
pts	The point value of the food.

Returns

The number of food items in the list after addition.

Reimplemented from element.Element.

4.6.3.4 pts() [1/2]

```
food.Food.pts (
            self)
```

Property getter for the point value.

Returns

The point value (`pts`) of the food.

4.6.3.5 pts() [2/2]

```
food.Food.pts (
            self,
            value)
```

Property setter for the point value.

Parameters

value	New point value for the food.

Exceptions

TypeError	if the point value is not an integer.

Clamps the point value between 1 and PTS_MAX. If the value is out of bounds, the default point value is used.

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

- food.py

4.7 map.Map Class Reference

Class representing a map of simulation.

Public Member Functions

- size (self)

    Property getter for the size of the map.
- size (self, value)

    Property setter for the size of the map.
- **rmdCoord** (cls, datas=[ ])

Static Public Attributes

- int **MAX_SIZE** = 500

  Maximum allowed size for the map.
- tuple **DEFAULT_SIZE** = (200, 150)

  Default size of the map.

Static Protected Attributes

- tuple **_size** = DEFAULT_SIZE

  Internal storage for the size.

4.7.1 Detailed Description

Class representing a map of simulation.

4.7.2 Member Function Documentation

4.7.2.1 size() [1/2]

```
map.Map.size (
                self)
```

Property getter for the size of the map.

Returns

A tuple representing the current size of the map.

```
Getter for the size.
```

4.7.2.2 size() [2/2]

```
map.Map.size (
                self,
                value)
```

Property setter for the size of the map.

Parameters

value	A tuple of two integers representing the new size.

Exceptions

ValueError	if the size is not a tuple of two integers.
TypeError	if either dimension is not an integer.
ValueError	if any dimension is out of bounds.

```
Setter for the size with validation.
```

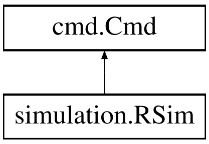
The documentation for this class was generated from the following file:

- map.py

4.8 simulation.RSim Class Reference

Class managing the main simulation loop and environment interactions.

Inheritance diagram for simulation.RSim:



Public Member Functions

- init (cls, numSave=0, size=Map.DEFAULT_SIZE)

  Initializes the simulation environment.
- generate (self, int nEntities, int nFoods)

  Generates a specified number of entities and food items.
- step (cls)

  Executes a single simulation step.
- run (cls)

  Runs the main simulation loop.

Public Member Functions inherited from cmd.Cmd

- startCmd (cls)
- stopCmd (cls)
- processCmd (cls, command)

  Processes commands entered in the terminal.

Public Attributes

- tuple _running = (Map.size[0] $*$ Map.size[1]) // 9

  Exit the simulation.

Public Attributes inherited from cmd.Cmd

- **input** = threading.Thread(target=cls._handle_input)
- int **save_duration** = 60

Static Public Attributes

- int **FPS_DEFAULT** = 180

  Default frames per second for the simulation.
- **save** = None

  Instance of the Save class for managing save operations.
- **visual** = None

  Instance of the Visual class for rendering the simulation.
- bool **mouse_clicking** = False

  Flag for mouse clicking state.
- int save_duration = 0

  Duration for displaying save messages.
- bool **verbose** = False

Static Public Attributes inherited from cmd.Cmd

- **lock** = threading.Lock()

  Lock for synchronizing access to terminal input.

Static Protected Attributes

- int **_fps** = FPS_DEFAULT

  Current frames per second setting.
- bool _running = False

  Flag indicating if the simulation is currently running.
- bool _pause = False

  Flag indicating if the simulation is paused.

Additional Inherited Members

Protected Member Functions inherited from cmd.Cmd

- _handle_input (cls)

  Handles terminal input for commands.
- _save (cls)

  Saves the current state of the simulation.
- _clear (cls)

  Clears the terminal screen and resets the input prompt.
- _spawn (cls, args)

  Spawns entities or food in the simulation.
- _help (cls)

  Displays the help message with available commands.

Protected Attributes inherited from cmd.Cmd

- bool **_running** = "help":

4.8.1 Detailed Description

Class managing the main simulation loop and environment interactions.

This class handles the initialization, running, and stepping of the simulation.

4.8.2 Member Function Documentation

4.8.2.1 generate()

```
simulation.RSim.generate (
              self,
             int nEntities,
             int nFoods)
```

Generates a specified number of entities and food items.

This class method creates the specified number of entities and food items by calling their respective `generate()` methods.

Parameters

nEntities	The number of entities to generate.
nFoods	The number of food items to generate.

- The method iterates over the range `nEntities` and generates an entity for each iteration.
- Similarly, it iterates over the range `nFoods` to generate food items.

4.8.2.2 init()

```
simulation.RSim.init (
              cls,
              numSave = 0,
              size = Map.DEFAULT_SIZE)
```

Initializes the simulation environment.

Parameters

numSave	Number of saves to initialize (default is 0).
size	Size of the map (default is Map.DEFAULT_SIZE).

Sets up the configuration, map size, and initializes save and visual components.

4.8.2.3 run()

```
simulation.RSim.run (
              cls)
```

Runs the main simulation loop.

Handles user input, updates simulation state, and renders visuals.

4.8.2.4 step()

```
simulation.RSim.step (
              cls)
```

Executes a single simulation step.

Moves all entities, checks their survival, and generates new food.

4.8.3 Member Data Documentation

4.8.3.1 _pause

```
bool simulation.RSim._pause = False  [static], [protected]
```

Flag indicating if the simulation is paused.

Execute a simulation step if not paused.

4.8.3.2 _running [1/2]

```
bool simulation.RSim._running = False  [static], [protected]
```

Flag indicating if the simulation is currently running.

Flag to indicate if the user is dragging the mouse.

Exit the simulation.

Store the last mouse position.

Temporarily store the pause state.

4.8.3.3 _running [2/2]

```
tuple simulation.RSim._running = (Map.size[0] * Map.size[1]) // 9
```

Exit the simulation.

Flag to indicate if the user is dragging the mouse.

Store the last mouse position.

Temporarily store the pause state.

4.8.3.4 save_duration

```
int simulation.RSim.save_duration = 0  [static]
```

Duration for displaying save messages.

Execute a simulation step if not paused.

Display pause screen.

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

- simulation.py

4.9 save.Save Class Reference

Class for saving and loading simulation states to and from binary files.

Public Member Functions

- None __init__ (self, numSave=0)

  Initializes the Save object.
- str __repr__ (self)

  Returns a string representation of the Save object.
- time (self)

  Gets the current simulation time.
- time (self, value)

  Sets the current simulation time.
- age (self)

  Gets the current age based on simulation time.
- save (self)

  Saves the current simulation state to a binary file.
- load (self)

  Loads the simulation state from a binary file.

Public Attributes

- int number = 0

  < Save file number
- **path** = Save._dir / (Save._name + str(self.number))
- **starting** = int(time())
- **last** = self._read_int(file, Save.NBYTES_TIME, 'file.last_loading')
- **time** = self._read_int(file, Save.NBYTES_TIME, 'file.sim_time')

Static Public Attributes

- int **NBYTES_TIME** = 8

  Number of bytes for time in the binary representation.
- int **NBYTES_COORD** = 2

  Number of bytes for coordinates in the binary representation.
- int **TIME_MAX** = 2 ∗∗ (NBYTES_TIME ∗ 8)

  Maximum time value.
- int **TIME_IN_AGE** = 40

Protected Member Functions

- _write_int (self, file, size, value, name="value")

  Writes an integer value to a binary file.
- _write_food (self, file)

  Writes the current food list to a binary file.
- _write_entity (self, file)

  Writes the current entity list to a binary file.
- _read_int (self, file, size, name="value")

  Reads an integer value from a binary file.
- _read_food (self, file)

  Reads food data from a binary file and populates the Food list.
- _read_entity (self, file)

  Reads entity data from a binary file and populates the Entity list.

Protected Attributes

- int **_time** = -1
- int **_energy** = 0
- int **_age** = 0

Static Protected Attributes

- **_dir** = path.PATH_SAVES

  Directory for save files.
- **_backup** = path.PATH_BACKUPS
- str **_name** = 'save_'

  Base name for save files.

4.9.1 Detailed Description

Class for saving and loading simulation states to and from binary files.

4.9.2 Constructor & Destructor Documentation

4.9.2.1 __init__()

```
None save.Save.__init__ (
              self,
              numSave = 0)
```

Initializes the Save object.

Parameters

numSave	An integer representing the save file number (default is 0).

4.9.3 Member Function Documentation

4.9.3.1 __repr__()

```
str save.Save.__repr__ (
              self)
```

Returns a string representation of the Save object.

Returns

A string containing save file path, last loading time, current time, and entities/food lists.

4.9.3.2 _read_entity()

```
save.Save._read_entity (
              self,
              file)  [protected]
```

Reads entity data from a binary file and populates the Entity list.

Parameters

file	The file object to read from.

4.9.3.3 _read_food()

```
save.Save._read_food (
              self,
              file)  [protected]
```

Reads food data from a binary file and populates the Food list.

Parameters

file	The file object to read from.

4.9.3.4 _read_int()

```
save.Save._read_int (
              self,
              file,
              size,
              name = "value")  [protected]
```

Reads an integer value from a binary file.

Parameters

file	The file object to read from.
size	Number of bytes for the integer.
name	Optional parameter for the value's name (used for error messages).

Returns

The integer value read from the file.

Exceptions

ValueError	if the value cannot be read correctly.

4.9.3.5 _write_entity()

```
save.Save._write_entity (
              self,
              file)  [protected]
```

Writes the current entity list to a binary file.

Parameters

file	The file object to write to.

4.9.3.6 _write_food()

```
save.Save._write_food (
              self,
              file)  [protected]
```

Writes the current food list to a binary file.

Parameters

file	The file object to write to.

4.9.3.7 _write_int()

```
save.Save._write_int (
              self,
              file,
              size,
              value,
              name = "value")  [protected]
```

Writes an integer value to a binary file.

Parameters

file	The file object to write to.
size	Number of bytes for the integer.
value	The integer value to write.
name	Optional parameter for the value's name (used for error messages).

Exceptions

ValueError	if value does not fit in the specified byte size.

4.9.3.8 age()

```
save.Save.age (
              self)
```

Gets the current age based on simulation time.

Returns

The calculated age as an integer.

4.9.3.9 load()

```
save.Save.load (
              self)
```

Loads the simulation state from a binary file.

Exceptions

IOError	if file operations fail.
ValueError	if the data in the file is invalid.

4.9.3.10 save()

```
save.Save.save (
              self)
```

Saves the current simulation state to a binary file.

Exceptions

IOError	if file operations fail.
ValueError	if a value exceeds its byte limits.

4.9.3.11 time() [1/2]

```
save.Save.time (
              self)
```

Gets the current simulation time.

Returns

The current time as an integer.

4.9.3.12 time() [2/2]

```
save.Save.time (
              self,
              value)
```

Sets the current simulation time.

Parameters

value	An integer representing the time to set.

Exceptions

TypeError	if the value is not an integer.
ValueError	if the value is out of range.

4.9.4 Member Data Documentation

4.9.4.1 number

```
int save.Save.number = 0
```

< Save file number

< Last loading timestamp

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

- save.py

4.10 visual.Visual Class Reference

Class that allows you to manage the visual aspect of the simulator.

Public Member Functions

- __init__ (self)

  Initializes the Visual class and sets up the display.
- close (self)

  Closes the Pygame window and quits the application.
- camera (self)

  Handles camera movement.
- handle_mouse_scroll (self, event)

  Handles mouse scroll events for adjusting cell size.
- handle_mouse_drag (self, event)

  Handles mouse drag events to move the camera.
- show (self)

  Displays the entire visual representation of the simulation.
- pause (self)

  Displays a pause message on the screen.
- save (self, duration)

  Displays a saving message on the screen.

Public Attributes

- **screen** = pygame.display.set_mode((Config.visual.window_width, Config.visual.window_height), pygame.↩ RESIZABLE)
- **clock** = pygame.time.Clock()
- **camera_x** = DEFAULT_CAMERA[0]
- **camera_y** = DEFAULT_CAMERA[1]
- **cell_size** = DEFAULT_CELL_SIZE
- bool **dragging** = False
- tuple **last_mouse_pos** = (0, 0)
- int **time** = 0
- int **cell_size** = 5:

Protected Member Functions

- _draw_background (self)

  Draws the background of the simulation.
- _draw_foods (self)

  Draws food items on the screen.
- _draw_entities (self)

  Draws entities on the screen.

4.10.1 Detailed Description

Class that allows you to manage the visual aspect of the simulator.

4.10.2 Constructor & Destructor Documentation

4.10.2.1 __init__()

```
visual.Visual.__init__ (
              self)
```

Initializes the Visual class and sets up the display.

This constructor initializes Pygame, sets the window size and caption, and initializes camera and cell size variables. It also prepares for mouse dragging events.

4.10.3 Member Function Documentation

4.10.3.1 _draw_background()

```
visual.Visual._draw_background (
              self)  [protected]
```

Draws the background of the simulation.

This method fills the screen with the background color and draws the boundaries of the map.

4.10.3.2 _draw_entities()

```
visual.Visual._draw_entities (
              self)  [protected]
```

Draws entities on the screen.

This method iterates through the list of entities and draws them at their respective positions.

4.10.3.3 _draw_foods()

```
visual.Visual._draw_foods (
              self)  [protected]
```

Draws food items on the screen.

This method iterates through the list of food items and draws them at their respective positions.

4.10.3.4 camera()

```
visual.Visual.camera (
              self)
```

Handles camera movement.

This method is currently empty and can be filled in with logic to manage camera movement based on user input, allowing for free movement.

4.10.3.5 close()

```
visual.Visual.close (
              self)
```

Closes the Pygame window and quits the application.

This method should be called to properly close the Pygame environment and release resources.

4.10.3.6 handle_mouse_drag()

```
visual.Visual.handle_mouse_drag (
              self,
              event)
```

Handles mouse drag events to move the camera.

This method updates the camera position based on mouse movement while dragging with the left mouse button.

Parameters

event	The Pygame event to handle.

4.10.3.7 handle_mouse_scroll()

```
visual.Visual.handle_mouse_scroll (
              self,
              event)
```

Handles mouse scroll events for adjusting cell size.

This method modifies the cell size based on mouse wheel scroll events.

Parameters

event	The Pygame event to handle.

4.10.3.8 pause()

```
visual.Visual.pause (
              self)
```

Displays a pause message on the screen.

This method renders a pause message when the simulation is paused.

4.10.3.9 save()

```
visual.Visual.save (
              self,
              duration)
```

Displays a saving message on the screen.

This method renders a "Saving..." message during save operations, positioned at the bottom right corner of the window.

Parameters

duration	The duration of the saving operation.

Returns

The duration parameter for potential use elsewhere.

4.10.3.10 show()

```
visual.Visual.show (
              self)
```

Displays the entire visual representation of the simulation.

This method calls the background, food, and entity drawing methods to render the current state of the simulation on the screen.

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

- visual.py

Chapter 5

File Documentation

5.1 cmd.py File Reference

File containing the class *Cmd*.

Classes

- class cmd.Cmd

  Class for handling terminal commands and interaction in the simulation.

5.1.1 Detailed Description

File containing the class *Cmd*.

Date

2024-10-05

Author

Rabyte Studio

5.2 config.py File Reference

File containing the class *Config*.

Classes

- class config._Config_Visual
- class config.Config

Variables

- str **config.DEFAULT_WINDOW_NAME** = 'RSim - Simulation'
- int **config.DEFAULT_WINDOW_WIDTH** = 800
- int **config.DEFAULT_WINDOW_HEIGHT** = 600
- **config.c** = Config()

5.2.1 Detailed Description

File containing the class *Config*.

Date

2024-10-05

Author

Rabyte Studio

5.3 element.py File Reference

File containing the class *Element*

Classes

- class element.Element

  Class representing an element on the map.

Variables

- **element.size**

5.3.1 Detailed Description

File containing the class *Element*

Date

2024-10-03

Author

Rabyte Studio

5.4 entity.py File Reference

File containing the class *Entity*.

Classes

- class entity.Entity

  Class representing an entity that extends Element and includes properties for energy, time, and age.

Variables

- **entity.size**

5.4.1 Detailed Description

File containing the class *Entity*.

Date

2024-10-03

Author

Rabyte Studio

5.5 food.py File Reference

File containing the class *Food*

Classes

- class food.Food

  Class representing food objects in the simulation.

Variables

- **food.size**

5.5.1 Detailed Description

File containing the class *Food*

Date

2024-10-04

Author

Rabyte Studio

5.6 map.py File Reference

File containing the class *Map*

Classes

- class map.Map
  Class representing a map of simulation.

Variables

- **map.size**

5.6.1 Detailed Description

File containing the class *Map*

Date

2024-10-03

Author

Rabyte Studio

5.7 save.py File Reference

File containing the class *Save*.

Classes

- class save.Save
  Class for saving and loading simulation states to and from binary files.

Variables

- **save.size**
- **save.save** = Save(numSave=1)

5.7.1 Detailed Description

File containing the class *Save*.

Date

2024-10-04

Author

Rabyte Studio

5.8 simulation.py File Reference

File containing the class *RSim*, responsible for running the simulation of the environment.

Classes

- class simulation.RSim

  Class managing the main simulation loop and environment interactions.

5.8.1 Detailed Description

File containing the class *RSim*, responsible for running the simulation of the environment.

File containing the class *Visual*.

Date

2024-10-04

Author

Rabyte Studio

Date

2024-10-05

Author

Rabyte Studio

Index