Colony Module

Modules that use Colony Module:

* UI

Modules Used by Colony Module

* Harvest
* Map
* Control
* Core

Members:

* Colony manager
* WorkerPool

Scripts

Colony Manager

# Variables:

Static:

* Public Instance: used for singleton

Public:

* Storage (get private set): script handler of food
* MaxPopulation(get)
* foodRequirement (get)
* Action onPopulationChange
* AvailableWorkers: alive workers that can perform actions
* AvailableAnts: alive ants including followers and player
* BuildingAnts: ants busy building in the anthill
* FollowerAnts: ants following the player
* CurrentPopulation: all alive ants in the nest
* AvailableFood: food in the storage

editor:

* worker stats
* worker prefab
* starting ants

private

* max population
* List AllAntsList
* List workersList
* List followersList
* buildingAntsRemaining: stores worker ants called to build but not yet in the nest
* buildingAntsCollected: stores worker ants building in the nest
* restingAnts: stores ants resting before going out and explore
* Player Reference
* Leader reference
* oneSecondCounter
* timeWithoutfood time in seconds since there is no food in the nest

Unity Methods

* Awake:
  + instance, storage, player and leader assignation
  + initialize workerpool
  + create starting workers (uses startingAnts)
  + calls onPopulationChange
* Update:
  + Every second calls FeedAnts()

Private methods

* FeedAnts: calls storage.Consume() with foodRequirement. If food runs out increase timewithoutfood, when is greater than workeStats.health kills one ant every second. Death priority is as follows:
  + First kill resting ants;
  + Then building ants
  + Then workers/followers from younger to older
  + If no more ants damage player 1/second;
* StartBuildingProcess deactivates needed ants starting close to nest. Updates buildingRemaining, buildingCollected and restingAnts. And calls to reactivation after build time.
* ReactivateBuilders: coroutine, after buildTime activates collectedbuildingAnts. Updates buildingRemaining and buildingCollected, and can handle multiple building processes at the same time. Calls onPopulationChange
* ReactivateRestingAnt: after restTime activates one ant and updates RestingAnts
* CreateWorker: gets a worker from the pool, adds the listeners for enter anthill and death. Teleports it to the nest and updates AllAnts and workers Lists. Calls onPopulationChange
* RemoveAnt: removes ant from allAnts worker and/or follower lists. If a worker removes listeners. Returns worker to pool. Calls onPopulationChange

Listeners:

* OnEnterAntill: Action passed to workerController EnterAnthill. Deactivates worker and updates buildingCounters or restingAnts.
* OnPlayerDeath: if there are worker ants play replace worker for player and keep playing if not GameOver. Returns to main menu
* GetOnWorkerDeathListener: returns a listener for that worker

Public methods

* Apply cost. Consumes required food and if needed call StartBuildingProcess
* IncreaseMaxPopulation increase maxPopulation and calls onPopulationChange
* IncreaseMaxFollowers; leader. increaseFollowers
* AddWorker; if not overpopulated createWorker
* IncreaseMaxHealth updates workerStats
* IncreaseMaxDamage updates workerStats
* IncreaseMaxSpeed updates workerStats
* Increase carryCapacity updates workerStats