GAMA

Sommario

[1.INTRODUCTION 1](#_Toc107947083)

[TERMINOLOGIA 2](#_Toc107947084)

[Agent-oriented programming 2](#_Toc107947085)

[Software agent 2](#_Toc107947086)

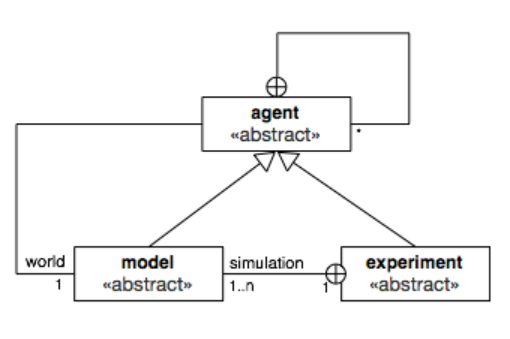
[RIFERIMENTI 3](#_Toc107947087)

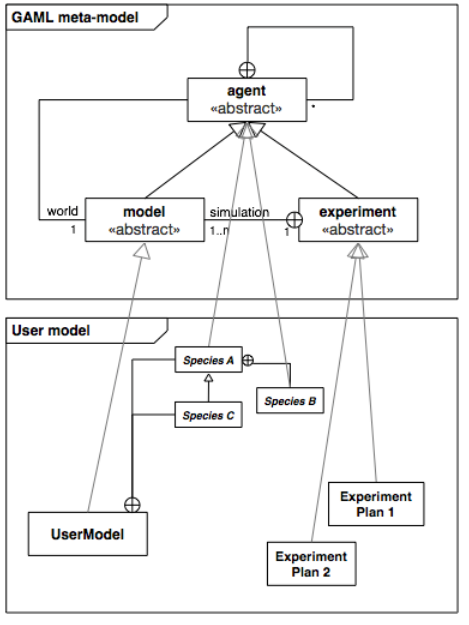
# 1.INTRODUCTION

GAML is an **agent-oriented language** dedicated to the definition of agent-based simulations. It takes its roots in object-oriented languages like Java or Smalltalk, but extends the object-oriented programming approach with powerful concepts (like skills, declarative definitions or agent migration) to allow for a better expressivity in models.

The role of GAML is to support modelers to write their **models**, which are specifications of **simulations** that can be executed and controlled during **experiments,** themselves specified by **experiment plans.**

Like in the object-oriented paradigm, where the notion of class is used to supply a specification of objects, agents in GAML are specified by their **species** which provides them with a set of **attributes (what they know), actions (what they can do), behaviors (what they actually do)** and also specifies properties of their **population,** for istance its **topology (how they are connected)** or **schedule (in which order and when they should execute).**

****



# 2.ORGANIZATION OF A MODEL

## 2.1.Model header(model species)

Declaration name of the model



Importing a model can take 2 forms. The first one, called **inheritance import,** in which all the declarations of the model imported will be merged with those of the current model.The second one, called **usage import,** reserved for **using models as micro-models of the current model.**

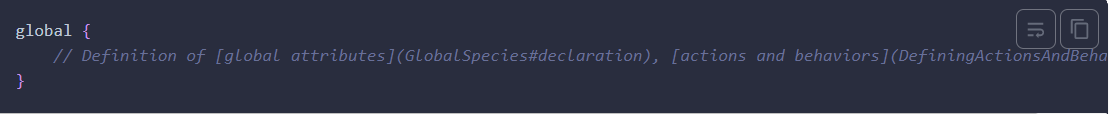
Inheritance import

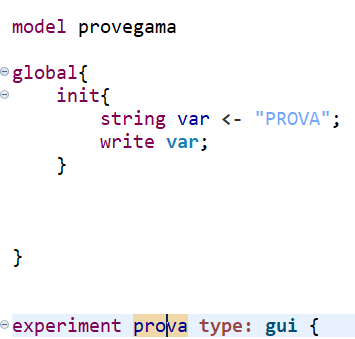


Usage import

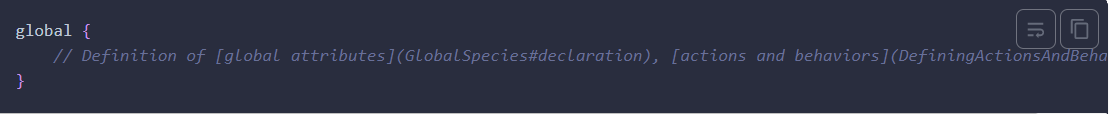


The last part of the **header** is the definition of the global species:



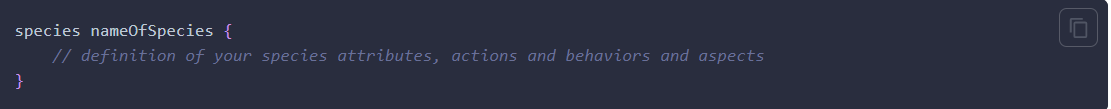


## 2.2.Species declaration

The **special species global** is the world species. You will declare here all the global attributes/actions/behaviors.

**Regular species** can be declared with the keyword **species.** You can declare several regular species, and they all have to be named.

A species defines its **attributes, actions and behaviors** and **aspects.**

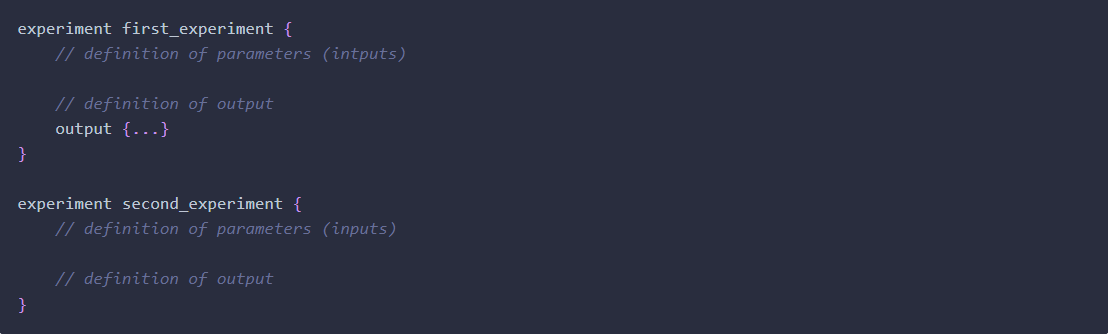
****

Species position:

**Immagine che contiene testo

Descrizione generata automaticamente**

## 2.3.Experiment declarations



### 2.3.1.GUI Experiment

GUI experiment allows you to display a **graphical interface**

****

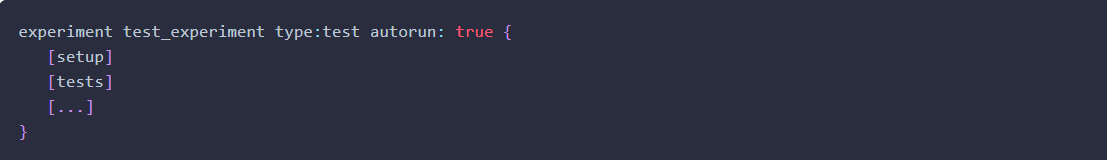
### 2.3.2.BATCH Experiment

**Batch experiment** allows you to execute numerous successive simulation runs (often used for model exploration). It is declared with the following structure:



### 2.3.3.Test experiment

**Test experiment** allows you to write unit tests on a model (used to ensure its quality). It is declared with the following structure:

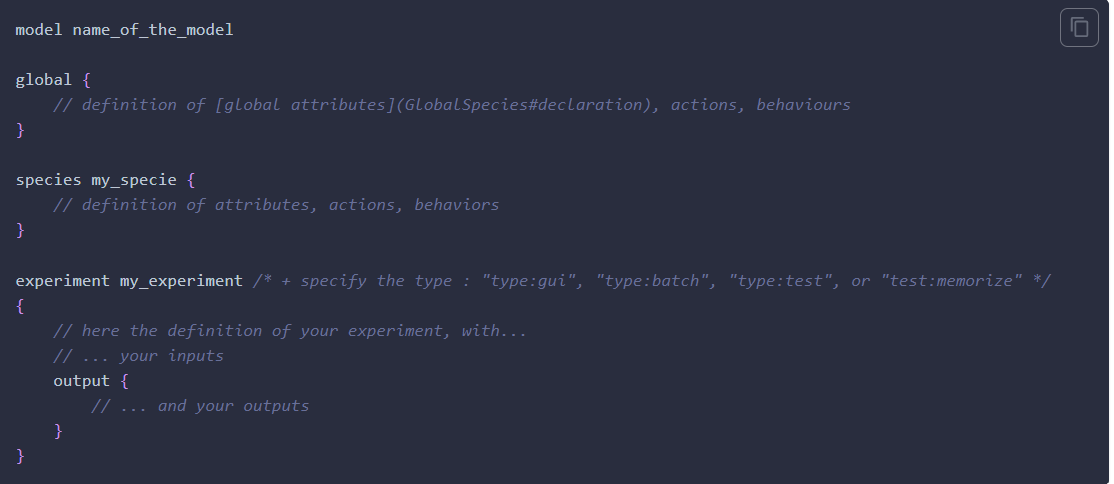


### 2.3.4.Memorize experiment

**Memorize experiment** allows you to store each step of simulation in memory and to backtrack to previous steps. Is is declared with the following structure:



## 2.4.Basic skeleton of a model



# 3.BASIC PROGRAMMING CONCEPTS IN GAML

## 3.1.Variables

### 3.1.1.Basic types

### 3.1.2.The point type

### 3.1.3.A word about dimensions

## 3.2.Declare variables using facet

### 3.3.Operators in GAMA 3.3.1.Matematical operators

### 3.3.2.Logical operators

### 3.3.3.Comparison operators

### 3.3.4.Type casting operators

### 3.3.5.Other operators

## 3.4.Loop

## 3.5.Manipulate containers

## 3.6.Random values

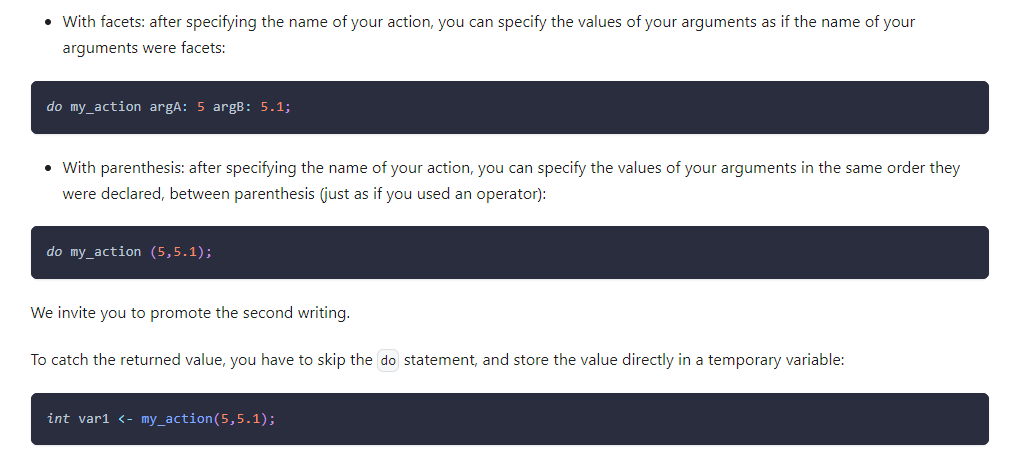
# DEFINING ACTIONS AND BEHAVIORS

## Declare an action

An **action** is a **function or procedure** run by an instance of species. An action can return a value (in that case, the type of return has to be specified just before the name of the action), or not (in that case, you just have to put the keyword action before the name of the action).



## Call an action

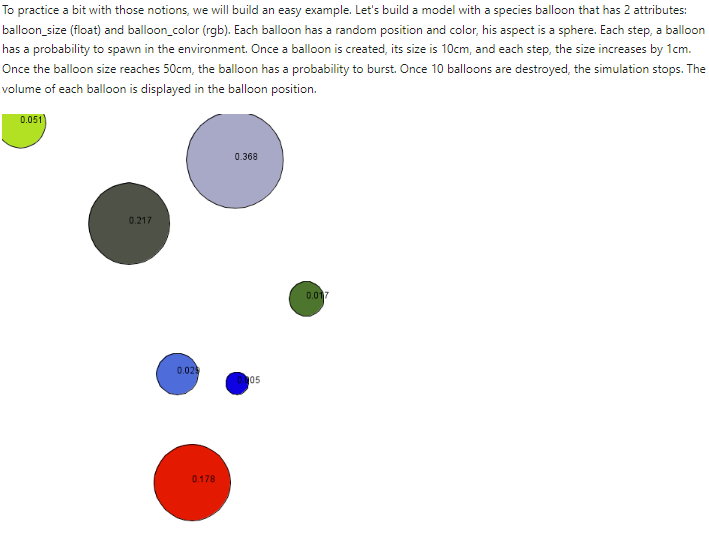


## Behavior

A **behavior** or **reflex** is a set of statements which is called **automatically** at each time step by an agent.Note that, a behavior is linked to an **architecture;** the reflex based architecture is the default one, others can be used with the controls facet of the species.



## Example





# INTERATION BETWEEN AGENTS

## The ask statement

The **ask statement** can be used in any **reflex** or **action** scope.It is used to specify the interaction between the instances of your species and the other agents.You only have to specify the species of the agents you want interact with.

# ATTACHING SKILLS

**Skills** are built in modules that provide a set of related **built in attributes and built in actions** (in addition to those already proposed by GAMA) to the species that declare them.

# CONTROL ARCHITECTURE

GAMA allows the modeller to attach built in control architecture **to agents**.

## Finite state machine

**FSM (Finite State Machine)** is a finite state machine-based behavior model. During its life cycle, the agent can be in several states. At any given time step, it is one single state. Such an agent needs to have one single state (the state in which it will be at its initialization).

## Task-Based

**Task-based**

## Rules-based architecture

## User control architecture

# USING EQUATIONS(molto interessante…da integrare quando ho tempo)

# TERMINOLOGIA

## Agent-oriented programming

**Agent oriented programming** is a programming paradigm where the costruction of the software is centered on the concept of **software agents**.

**Agent oriented programming** can be viewed as a specialization of **object oriented programming.**

The **state of an agent** consists of components such as beliefs, decisions, capabilities, and obligations, for this reason the state of an agent is called its **mental state.**

## Software agent

In computer science a **software agent** is a computer program that acts for a user or other program in a relationship of agency, which derives from the Latin agere(to do). **Agents** are colloquially known as **bots** or **robots.**

They may be embodied, as when execution is paired with a robot body, or as software such as chatbot executing on a phone (Siri) or other computer device. Software agents may be **autonomous** or **work together** with other agent or people.

# RIFERIMENTI

[1] [Documentazione ufficiale di GAMA](https://gama-platform.org/)

[2] [Wikipedia: agent oriented programming](https://en.wikipedia.org/wiki/Agent-oriented_programming)

[3] [Wikipedia: software agent](https://en.wikipedia.org/wiki/Software_agent)

[4] [sciencedirect: agent oriented programming](https://www.sciencedirect.com/science/article/abs/pii/0004370293900349)