



artisoc Cloudレシピブック

10. artisoc4からartisoc Cloudへ移行しよう

(株) 構造計画研究所
創造工学部

<https://mas.kke.co.jp>

artisoc4で作成したモデルをartisoc Cloudに変換する手順を解説します

- artisoc Cloudの紹介
- 『MASの教材』の紹介
- モデルの実行画面
- モデルツリーの定義
- 出力設定の定義
- エージェントルールの書き方
- 関数対応表
- マップの初期値設定は描画ツール
- 学びのための情報提供



[飛ぶ鳥モデルからボイドモデルへ](#)

artisoc Cloudはartisoc4と同じ、ユーザフレンドリーなマルチエージェント・シミュレータです。

- ① pythonでモデルが組める（学習が容易、ライブラリが使える）
- ② Webブラウザで利用できる（インストール不要、どこでも）
- ③ 実行結果を視覚的かつ直感的に把握できる（すぐに可視化）



『MASの教材』の紹介

『MASの教材』は、MASを使った授業のために作成したコンテンツです。
artisoc Cloudとartisoc4の両方のコンテンツが含まれています。
artisoc4のモデルをartisoc Cloudへ移行する際は参考にしてください。



MASの教材

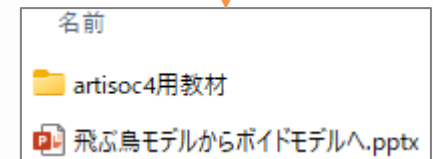
「詳しく見る」をクリック



「コンテンツ式のダウンロード」
をクリック



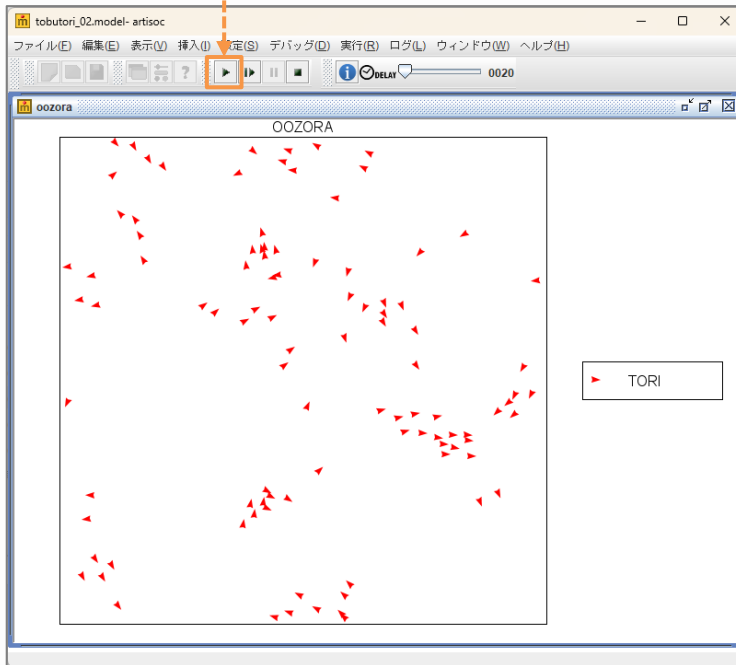
tobutori.zip



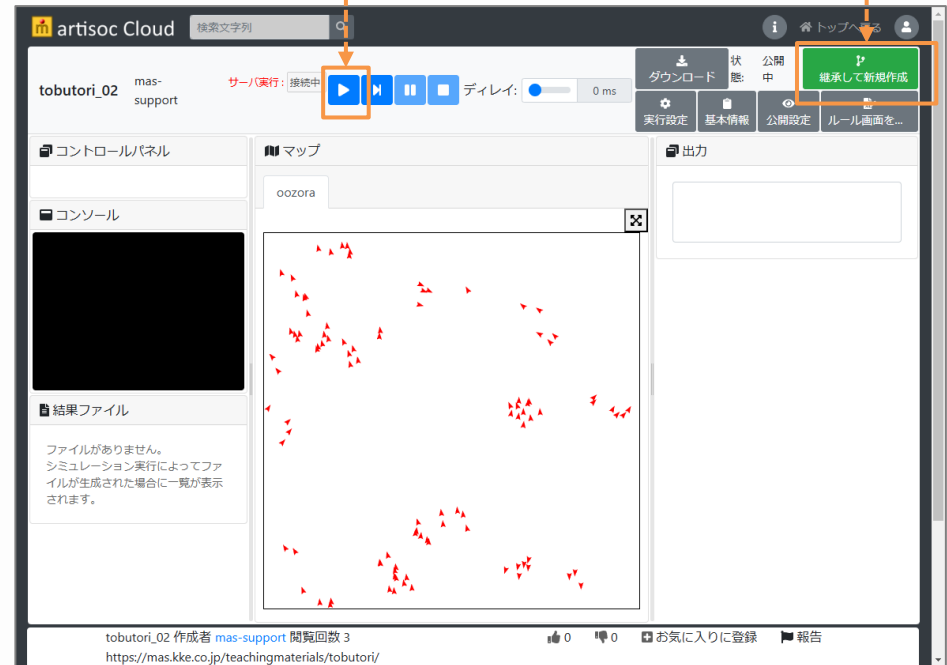
「飛ぶ鳥モデルからボイドモデルへ」のモデルを実行すると、同じように動作します。

モデルの実行ボタン

公開されているモデルを編集するときは、
「継承して新規作成」をクリック



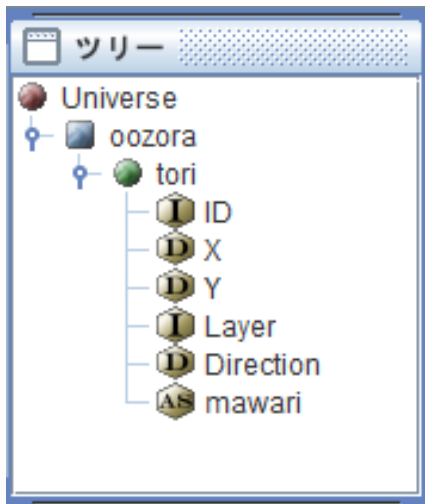
artisoc4『tobutori_02.model』



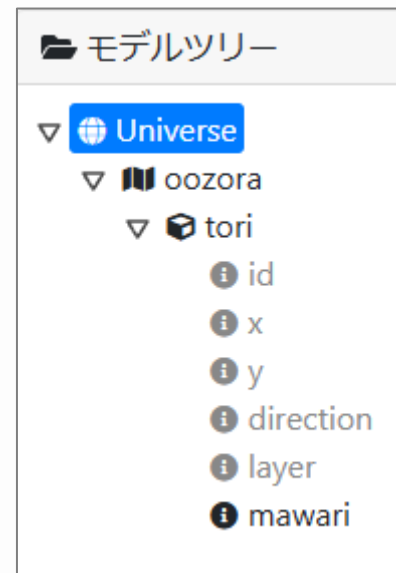
artisoc Cloud『tobutori_02』

モデルツリーはほぼ同じです。

artisoc4では変数を追加するときに変数名と変数の型を指定します。
artisoc Cloudでは変数名のみ指定します。



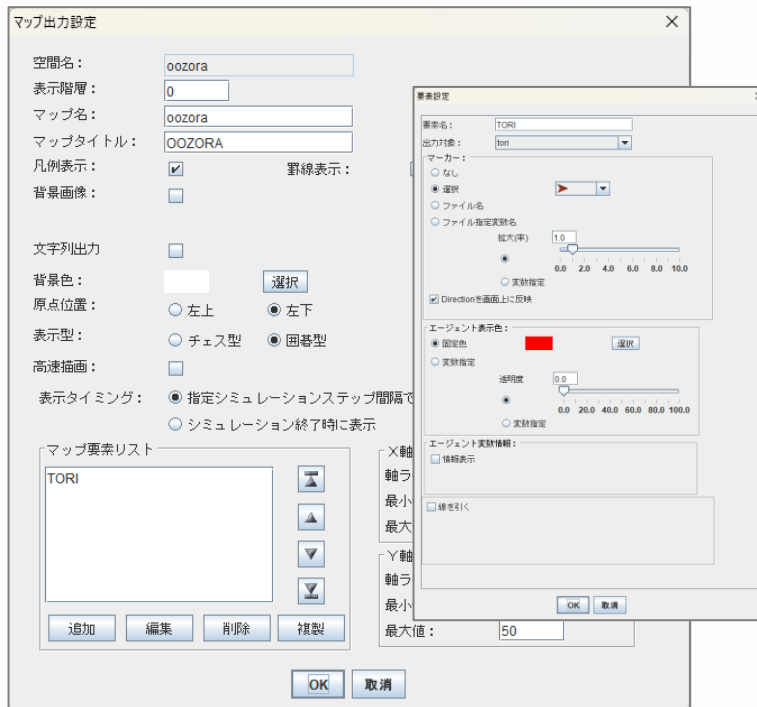
artisoc4



artisoc Cloud

出力設定はほぼ同じです。

設定方法は『[レシピブック 07. 出力設定をマスターしよう](#)』を参照してください。



artisoc4



artisoc Cloud

エージェントルールの書き方はかなり違います。

artisoc Cloud python
artisoc4 VisualBasicライクな言語

まずは『[ルール文法](#)』を読んで、pythonの基本的な書き方を理解しましょう。

ルール文法

概要

本意では、artisoc Cloud モデルのルール記述についての仕様を説明します。モデルのルールは、基本的に Python 言語（ver.3 系）を用いて記述します。

Python の基礎文法

artisoc Cloud では、プログラミング言語 Python を用いてモデルのルールを記述します。本節では、artisoc Cloud のルール記述に最低限必要な Python の基礎的な文法を説明します。詳しくは Python の公式ドキュメントを参照してください。なお、一部に artisoc Cloud 特有の内容も含まれていますが、Python 使用経験のある方は読み飛ばしても問題ありません。

変数・代入・演算

変数とは、値を保存しておく箱のようなものです。たとえば以下のように記号「=」を用いて記述することで、変数 a を作成しそこに値 3 を保存することができます。これを代入と呼びます。

```
a = 3
```

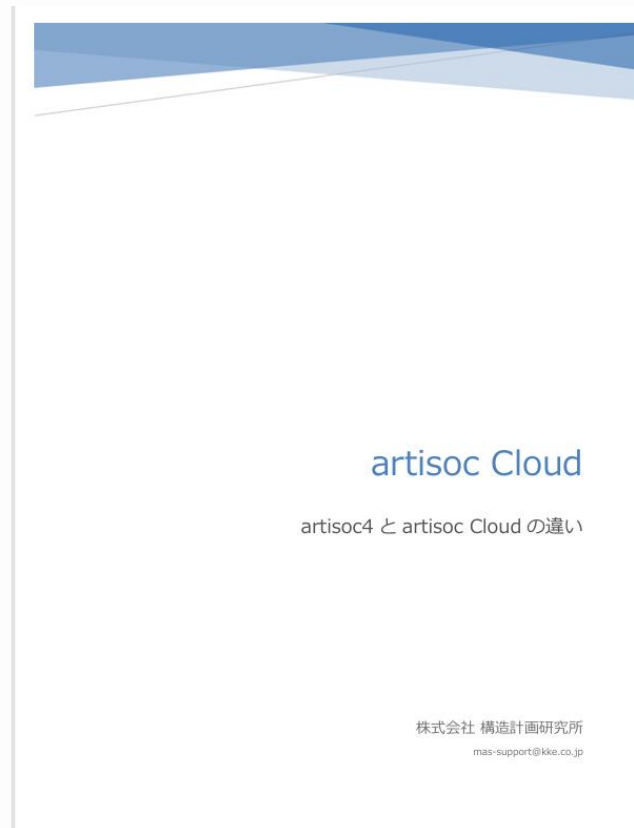
ここで記号「=」は右辺の値を左辺に代入する操作を表します。数学とは異なり、両辺が等しいことを示す記号ではないことに注意してください。

変数名は基本的には自由につけることができますが、以下のようなルールがあります。

- 大文字と小文字は区別される。
- 変数名に空白を含んではいけない。

71

『artisoc4とartisoc Cloudの違い』をまとめています。
次ページ以降、特に注意する点について解説します。



「artisoc4とartisoc Cloudの違い」

- 改行とインデント

artisoc4

artisoc4では、改行とインデントについてのルールはありません。つまり、改行とインデントはコードを見やすくするために挿入するものであって、改行とインデントがなくともコードは問題なく動作します。

```
//改行とインデントを入れた見やすいコード
Agt_Init{
  My.X = 3
  My.Y = 3
}
```

artisoc Cloud

artisoc Cloudでは、改行とインデントを正しく用いないとエラーになります。

とても重要！

```
# 正しいコード
def agt_init(self):
    self.x = 3
    self.y = 3
```

```
# エラーになるコード
def agt_init(self):
    self.x = 3
    self.y = 3 # インデントが揃っていないためエラー
```

- if文

artisoc4

artisoc4ではif文は以下のように記述します。

```
If [条件文1] Then  
    [処理1]  
ElseIf [条件文2] Then  
    [処理2]  
Else  
    [処理3]  
End If
```

※Elseif, Elseは省略可。Elseifは複数書くことも可。

artisoc Cloud

※ if文やfor文の行末には「:」を書きます

artisoc Cloudでは、if文は以下のように記述します。処理はインデント内に記述することに注意します。

```
if [条件文1]:  
    [処理1]  
elif [条件文2]:  
    [処理2]  
else:  
    [処理3]
```

※elif, elseは省略可。elifは複数書くことも可。

- 配列とリスト

artisoc Cloud ※ 配列はリスト型や辞書型（key, value）を使います

artisoc Cloudにおいては、artisoc4での配列に正確に対応するデータ型は存在しません。その代わりに、**リスト型**の変数を用いることができます。

リスト型の変数を定義するには、下のように `[]` を用います。

```
Universe.list1 = [1, 2, 3, 4, 5] # 長さ5のリストを定義
```

リスト型の変数にアクセスするには、`[]` を用いて要素番号を指定します。

```
Universe.list1[3] = 10 # リストの3番目の要素に10を代入  
print(Universe.list1) # --> [1, 2, 3, 10, 5] (3番目の要素に10が入っている。要素番号は0から数えることに注意)
```

関数対応表 にartisoc4とartisoc Cloudの関数の対応をまとめています。

関数対応表

artisoc4 と artisoc Cloud の関数の対応についてまとめています。

引数など細かい仕様が異なる場合がありますので、詳しくは関数仕様を参照してください。

数値計算

artisoc4	artisoc Cloud
Abs	abs
Atn	atan
Cos	cos
DegreeToRad	radians
Exp	exp
Int	int
Log	log
NormInv	normalvariate
Pi	pi

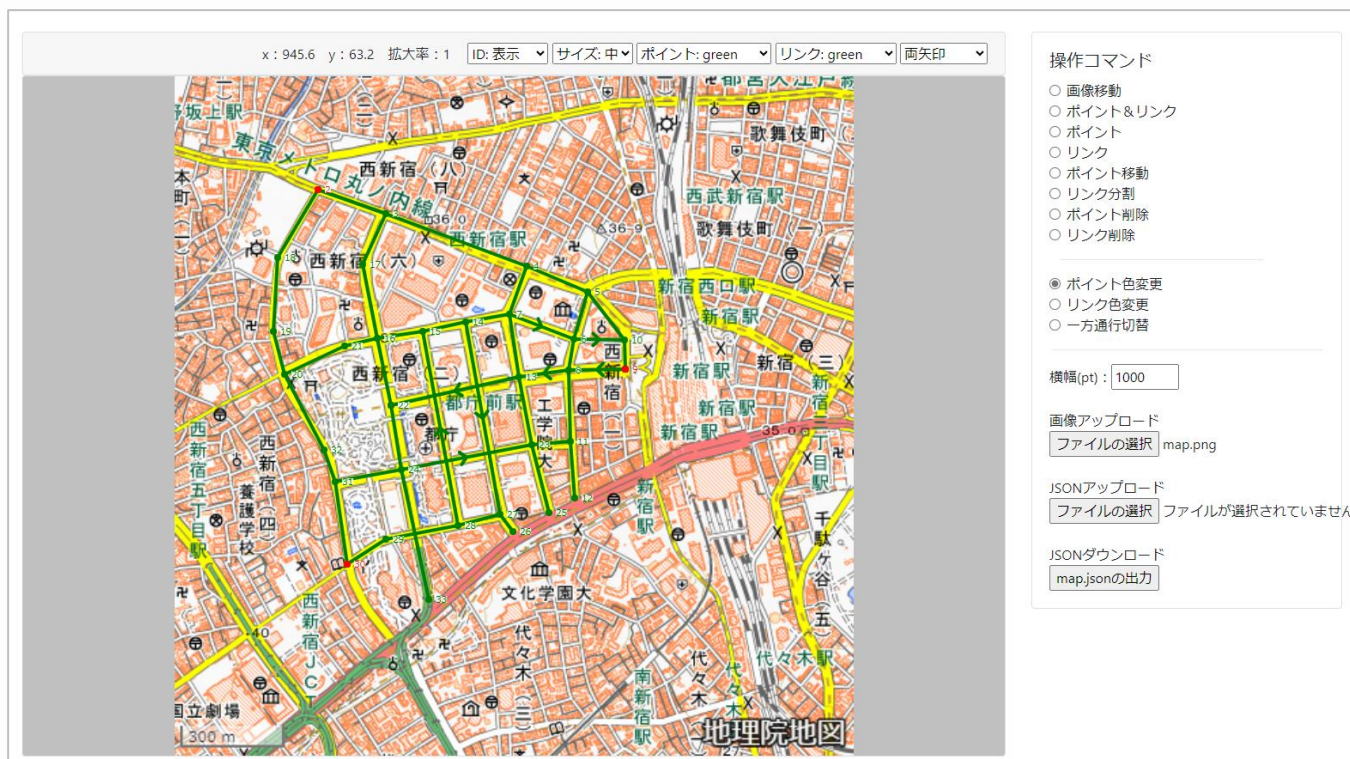
4

「関数対応表」

マップ上の初期値設定は、描画ツールを利用してください。

描画ツールの使い方は、

『[レシピブック 09.描画ツール2を使って歩くモデルの高速版](#)』を参照してください。



「描画ツール2」

Step1: 過去研究から着想を得る

- [MASのモデル](#) 身の回りの複雑系で紹介されているサンプルモデルを解説

Step2: モデリング講習を使って頭の中を整理する

- [artisoc モデリング講習](#) やりたいこと、つくりたいモデルを『文章』にする

Step3: artisoc Cloudの使い方を学ぶ

- [artisoc Cloud教科書](#) 書籍『人工社会構築指南』をベースに加筆
- [モデル作成のレシピブック](#) モデル作成のスキル習得のための指南書
- [質問掲示板](#) モデル作成で悩んだとき、ヒントが欲しいときの相談窓口

Step4: artisoc Cloudの学びの場

- [artisoc Cloud初級チュートリアル](#) 初級者向けのチュートリアル
- [artisoc Cloud勉強会](#) 勉強会に参加して仲間をつくろう

