Documentazione Kojo in italiano

Massimo Maria Ghisalberti - pragmas.org

2015-10-08

Indice

1 Kojo

Kojo è sviluppato da Lalit Pant (Himjyoti school, Dehradun - India) ed è utilizzato in varie scuole indiane, statunitensi, inglesi e svedesi. L'approccio usato nella piattaforma Kojo (http://www.kogics.net/kojo) è più ampio dei soliti ambienti per l'insegnamento. Può essere rivolto a più livelli di apprendimento ed è dotato di parti specifiche, per esempio per la sperimentazione in ambito matematico con un laboratorio basato su GeoGebra (http://www.geogebra.org/cms/it/). Il linguaggio utilizzato è Scala (http://www.scala-lang.org/). Scala è un linguaggio estremamente potente e multiparadigma (Orientato agli oggetti, funzionale) che può essere utilizzato a vari livelli, sufficientemente semplice nelle sue basi da poter essere insegnato in età scolare (dalla classe 4° primaria). La sua caratteristica di linguaggio funzionale lo fa particolarmente utile nella risoluzione di problemi matematici.

2 Documentazione funzioni in italiano

Alcune funzioni sono relative alla versione 2.4.09

Questo documento in pdf

Massimo Ghisalberti - tel.: +39 3939018290 – email: massimo.ghisalberti@gmail.com

2.1 Ambiente

- **pulisci()** - Pulisce l'area di disegno.
- **pulisciOutput()** - Pulisce l'area di output.

2.2 Tartaruga

- **nuovaTartaruga()** - Costruisce una nuova Tartaruga nel punto centrale.
- **nuovaTartaruga(x: Double = 0, y: Double = 0, costume: String = "/images/turtle32.png")** - Costruisce una nuova Tartaruga in base ai dati forniti.
- **rimuovi()** - Elimina la tartaruga.
- **visibile()** - Rende la tartaruga visibile.
- **invisibile()** - Rende la tartaruga invisibile.
- **salvaStile()** - Salva lo stile della tartaruga.
- **ripristinaStile()** - Ricarica lo stile della tartaruga salvato con *salvaStile()*.
- **assi()** - Mostra degli assi cartesiani sulla tartaruga.
- **rimuoviAssi()** - Elimina gli assi cartesiani dalla tartaruga.
- **indossaCostume(nomeDelFile: String)** - Imposta l'immagine della tartaruga.
- **indossaCostumi(nomeDelFile: String\)*** - Imposta le immagini della tartaruga.
- **indossaImmagine(immagine: java.awt.Image)** - Imposta l'immagine della tartaruga.
- **indossaImmaginei(immagini: java.awt.Image\)*** - Imposta le immagini della tartaruga.
- **prossimoCostume()** - Indossa il prossimo costume dalla lista delle immagini.
- **scalaCostume(fattore: Double)** - Scala la dimensione del costume.

2.3 Movimento

- **avanti()** - Sposta la tartaruga in avanti per 25 passi.
- **avanti(passi: Double)** - Sposta la tartaruga in avanti per il numero di passi fornito.
- **indietro(passi: Double)** - Sposta la tartaruga indietro per il numero di passi fornito.
- **destra(angolo: Double)** - Gira la tartaruga a destra dell'angolo fornito.
- **destra()** - Gira la tartaruga a destra di 90°.
- **sinistra(angolo: Double)** - Gira la tartaruga a sinistra dell'angolo fornito.
- **sinistra()** - Gira la tartaruga a sinistra di 90°.
- **saltaVerso(x: Double, y: Double)** - Fa saltare la tartaruga verso le coordinate fornite.
- **muoviVerso(x: Double, y: Double)** - Fa muovere la tartaruga verso le coordinate fornite.
- **cambiaPosizione(x: Double, y: Double)** - Cambia la posizione della tartaruga alle coordinate fornite.
- **salta()** - Fa saltare la tartaruga di 25 passi.
- **salta(n: Double)** - Fa saltare la tartaruga del valore fornito.

Massimo Ghisalberti - tel.: +39 3939018290 – email: massimo.ghisalberti@gmail.com

- **casa()** - Riporta la tartaruga nel punto di creazione.
- **verso(x: Double, y: Double)** - Fa puntare la tartaruga verso le coordinate fornite.
- **impostaDirezione(angolo: Double)** - Sposta la testa della tartaruga dei gradi forniti.
- **direzione** - Legge quale sia l'inclinazione della testa della tartaruga.
- **posizione** - Legge la posizione della tartaruga.
- **est()** - Posiziona la testa della tartaruga ad EST.
- **ovest()** - Posiziona la testa della tartaruga ad OVEST.
- **nord()** - Posiziona la testa della tartaruga ad NORD.
- **sud()** - Posiziona la testa della tartaruga ad SUD.
- **salvaPosizioneDirezione()** - Salva la posizione e la direzione della tartaruga.
- **ripristinaPosizioneDirezione()** - Ricarica la posizione e la direzione della tartaruga salvati con *salvaPosizioneDirezione()*.
- **ultimaLinea** - Ritorna le coordinate dell'ultima linea come Option.
- **ultimaSvolta** - Ritorna le coordinate dell'ultima svolta come Option.

2.3.1 Direzioni

- **Destra** - Tipo di dato per la direzione
- **Sinistra** - Tipo di dato per la direzione
- **Alto** - Tipo di dato per la direzione
- **Basso** - Tipo di dato per la direzione

2.4 Animazione

- **ritardo(n: Long)** - Rallenta il movimento della tartaruga.
- **valoreRitardo** - Legge il valore di ritardo della tartaruga.

2.5 Figure geometriche

- **arco(raggio: Double, angolo: Double)** - Disegna un arco dato il raggio.
- **arco2(raggio: Double, angolo: Double)** - Disegna un arco dato il raggio.
- **cerchio(raggio: Double)** - Disegna un cerchio dato il raggio.
- **punto(diametro: Int)** - Disegna un punto dato il diametro.
- **quadrato(passi: Double = 100, direzione: Direzione = Destra)** - Disegna un quadrato girando nella direzione fornita *Destra* o *Sinistra*.
- **triangolo(lato: Double, direzione: Direzione = Destra)** - Disegna un triangolo girando nella direzione fornita *Destra* o *Sinistra*.
- **superficie** - Legge la superficie della figura disegnata
- **perimetro** - Legge il perimetro della figura disegnata

Massimo Ghisalberti - tel.: +39 3939018290 – email: massimo.ghisalberti@gmail.com

2.6 Disegno

- **scrivi(t: Any)** - Fa scrivere del testo alla tartaruga.
- **impostaCarattere(font: java.awt.Font)** - Imposta il carattere di scrittura.
- **impostaGrandezzaCarattere(dimensione: Int)** - Imposta la dimensione del carattere di scrittura.
- **abbassaPenna()** - Abbassa la penna sull'area di disegno per disegnare.
- **alzaPenna()** - Alza la penna sull'area di disegno per smettere di disegnare.
- **èLaPennaAbbassata** - Legge se la penna è abbassata o no.
- **colorePenna(colore: Color)** - Imposta il colore della penna.
- **coloreRiempimento(colore: Color)** - Imposta il colore del riempimento.
- **impostaSpessorePenna(n: Double)** - Imposta lo spessore della penna.
- **sfondo(c: Color)** - Imposta il colore dello sfondo dell'area di disegno.
- **gradiente(c1: Color, c2: Color)** - Imposta il gradiente dello sfondo dell'area di disegno.

2.6.1 Colori predefiniti

- **blu** - Colore predefinito
- **rosso** - Colore predefinito
- **giallo** - Colore predefinito
- **verde** - Colore predefinito
- **porpora** - Colore predefinito
- **rosa** - Colore predefinito
- **marrone** - Colore predefinito
- **nero** - Colore predefinito
- **bianco** - Colore predefinito
- **senzaColore** - Colore predefinito

2.7 Audio

- **suona(voce: Voice)** - Esegue uno spartito Voice

2.8 Cicli

- **fai(fn: Tartaruga => Unit)** - Ripete il blocco di codice una volta, *self* è il riferimento alla tartaruga.
- **rifai(fn: Tartaruga => Unit)** - Ripete il blocco di codice 30 volte al secondo, *self* è il riferimento alla tartaruga.
- **ripeti(n: Int)(block: => Unit)** - Ripete il numero n di volte un blocco fornito.
- **ripetizione(n: Int)(block: Int => Unit)** - Ripete il numero n di volte un blocco fornito fornedo al blocco il valore corrente.
- **ripetiFinché(condizione: => Boolean)(block: => Unit)** - Ripete il blocco fornito finché la condizione è vera.
- **ripetiPerOgniElementoDi[T](sequenza: Iterable[T])(block: T => Unit)** - Ripete il blocco di codice per ogni elemento di una lista di elementi.

Massimo Ghisalberti - tel.: +39 3939018290 – email: massimo.ghisalberti@gmail.com

2.9 Condizioni

- **se(condizione) { blocco1 } altrimenti { blocco2 }** - Se *condizione* è vera sarà eseguito *blocco1* altrimenti *blocco2*.
- **seVero(condizione) {blocco }** - Se *condizione* è vera sarà eseguito *blocco1*.
- *espressione1* **oppure** *espressione2* - Se expressione1 è vera allora espressione1 altrimenti espressione2

2.9.1 operatori binari e ternari

- *espressione1* **?:** *espressione2* - Elvis Operator, simile a oppure ma lavora su valori nulli.
- *(condizione)* **??** *(blocco1)* **::** *(blocco2)* - Se *condizione* è vera sarà eseguito *blocco1* altrimenti *blocco2*.

2.10 Input e Output

- **leggiLinea(pronto: String = "")** - Legge una linea di testo in input.
- **scriviLinea(data: Any)** - Scrive il testo fornito.
- **scriviLinea()** - Scrive una linea di testo vuota.

2.11 Matematica

- **arrotonda(numero: Number, cifre: Int = 0): Double** - Arrotonda un valore al suo valore più prossimo.
- **numeroCasuale(limitiSuperiori: Int)** - Genera un numero casuale.
- **numeroDecimaleCasuale(limitiSuperiori: Int)** - Genera un numero decimale casuale.

2.11.1 tipi di dati

- **Intero** - Un numero intero.
- **Decimale** - Un numero decimale.
- **Stringa** - Un dato testuale.

2.12 Esempi

```
pulisci()

ritardo(200)
colorePenna(blu)
coloreRiempimento(verde)

ripeti(4) {
  avanti(100)
  destra(90)
}
```

```
pulisci()

ritardo(200)

colorePenna(gray)

var colore = Color(255, 0, 0, 150)

ripeti(15) {
  coloreRiempimento(colore)
```

Massimo Ghisalberti - tel.: +39 3939018290 – email: massimo.ghisalberti@gmail.com

```
  ripeti(4) {
    avanti(100)
    destra(90)
  }
  colore = hueMod(colore, 0.05)
  destra(360 / 15)
}
```

```
pulisci()

ritardo(20)

colorePenna(gray)

var colore = Color(255, 0, 0, 150)

ripeti(18) {
  coloreRiempimento(colore)
  ripeti(5) {
    avanti(100)
    destra(72)
  }
  colore = hueMod(colore, 0.05)
  destra(360 / 18)
}
```

```
pulisci()

def quadrato(t: Tartaruga, n: Int, delay: Int) {
    t.ritardo(delay)
    repeat(4) {
        t.avanti(n)
        t.destra()
    }
}

def occhi(t: Tartaruga, n: Int, delay: Int) {
    quadrato(t, n, delay)
    t.alzaPenna()
    t.avanti(n / 4)
    t.destra()
    t.avanti(n / 4)
    t.sinistra()
    t.abbassaPenna()
    t.coloreRiempimento(darkGray)
    quadrato(t, n / 2, delay)
}

val viso = nuovaTartaruga(-100, -100)
val occhioSinistro = nuovaTartaruga(-75, 25)
val occhioDestro = nuovaTartaruga(25, 25)
val bocca = nuovaTartaruga(-50, -50)
val naso = nuovaTartaruga(0, -25)
val capelli = nuovaTartaruga(-110, 100)
val corpo = nuovaTartaruga(25, -125)
val gambe = nuovaTartaruga(0, -150)

viso.fai { self =>
    self.coloreRiempimento(red)
    quadrato(self, 200, 200)
    self.invisibile()
}

occhioSinistro.fai { self =>
    self.coloreRiempimento(verde)
    occhi(self, 50, 800)
    self.invisibile()
}
```

Massimo Ghisalberti - tel.: +39 3939018290 – email: massimo.ghisalberti@gmail.com

```
43
44  occhioDestro.fai { self =>
45      self.coloreRiempimento(giallo)
46      occhi(self, 50, 800)
47      self.invisibile()
48  }
49
50  bocca.fai { self =>
51      self.ritardo(2000)
52      self.colorePenna(yellow)
53      self.impostaSpessorePenna(14)
54      self.destra()
55      self.avanti(100)
56      self.invisibile()
57  }
58
59  naso.fai { self =>
60      self.ritardo(4000)
61      self.colorePenna(yellow)
62      self.impostaSpessorePenna(20)
63      self.avanti(50)
64      self.invisibile()
65  }
66
67  capelli.fai { self =>
68      self.ritardo(200)
69      self.destra()
70      self.colorePenna(black)
71      self.impostaSpessorePenna(30)
72      self.avanti(220)
73      self.sinistra()
74      repeat(10) {
75          self.avanti(25)
76          self.indietro(25)
77          self.sinistra()
78          self.avanti(22)
79          self.destra()
80      }
81      self.avanti(25)
82      self.invisibile()
83  }
84
85  corpo.fai { self =>
86      self.coloreRiempimento(yellow)
87      self.cerchio(25)
88      self.invisibile()
89  }
90
91  gambe.fai { self =>
92      self.ritardo(3000)
93      self.colorePenna(black)
94      self.impostaSpessorePenna(30)
95      self.destra(180)
96      self.salvaPosizioneDirezione()
97      self.destra(30)
98      self.avanti(30)
99      self.ripristinaPosizioneDirezione()
100     self.sinistra(30)
101     self.avanti(30)
102     self.invisibile()
103 }
```

Massimo Ghisalberti - tel.: +39 3939018290 – email: massimo.ghisalberti@gmail.com