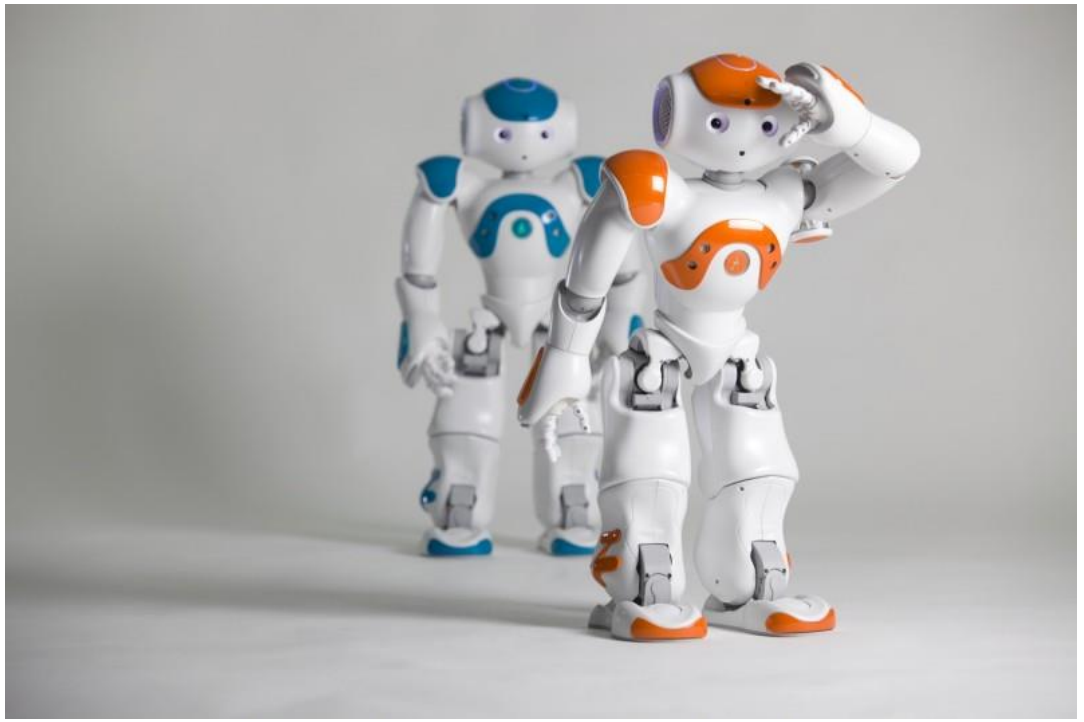
EPSI l'École d'ingénierie informatique

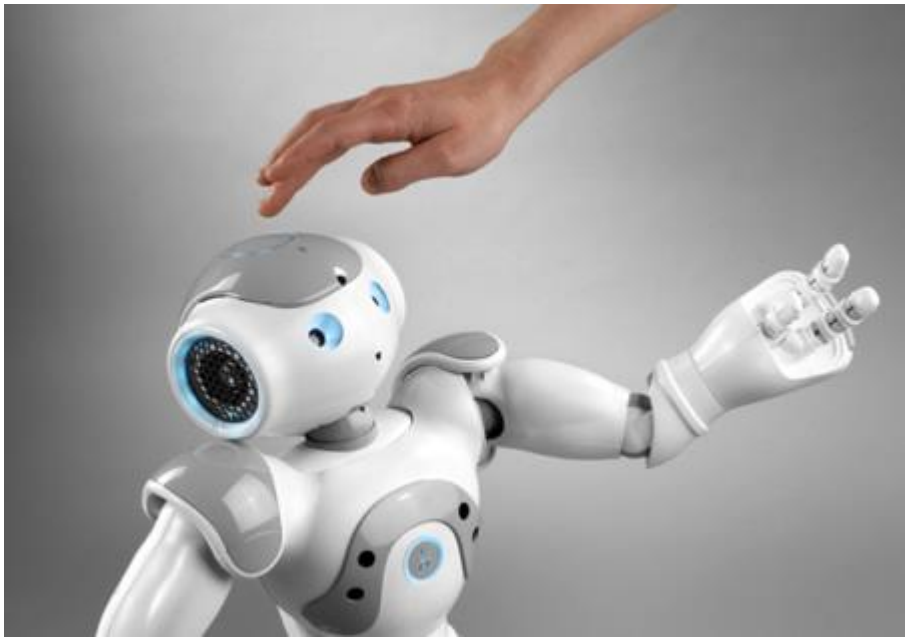


Support 2 : Python

Programmation Python dans Choregraphe

Programme de la séance

- Programmation Python dans Choregraphe
- Syntaxe Python
- Syntaxe des boîtes
- Exemple
- Application 1
- Utilisation des API
- Application 2



Python dans Choregraphe

- Les boîtes sont limitées
  - En nombre
  - En fonctionnalités
- Il est intéressant de créer ses propres boîtes
- Ajouter du code « maison » est rapidement indispensable
- Nao peut être programmé dans différents langages via des SDK, mais uniquement en Python dans Choregraphe

Syntaxe de Python - 1

- Types de base : int, float, bool (True ou False), str (" ou ')
- Conteneurs : list [], tuple (), set {}, dict {cle:valeur}
- Commentaire : # comm
- Opérateurs booléens : and, or, not
- Structures de contrôles :

```
if test:
   instructions
elif test :
   instructions
…
else:
   instructions
```

```
while condition:
   instructions
```

```
For variable in sequence:
   instructions
```

```
For var in range(deb,fin):
   instructions
```

Syntaxe de Python - 2

- Importation de bibliothèques : import Xxxx

- Création de classe :

class Nom(base)

- Création de méthodes :

```
def Nom(arg1, args2…) :
    …
    return xxx
```

Syntaxe particulière dans Choregraphe

- Tous les blocs correspondent à 1 classe
- Ils héritent de GeneratedClass

- Les attributs sont créés à l'initialisation dans la méthode onLoad(self) : self.nom = valeur

- Méthodes d'entrées (param : entrée reçue) :

def onInput_nomEntrée(self, param)

- Envoyer une valeur sur une sortie :

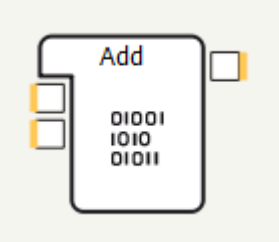
self.nomSortie(valeur)

Exemple : addition de deux nombres

- La boîte accepte deux entrées, puis envoie la somme en sortie

- 1ère étape : création de la boîte
  - 2 entrées numériques : number1 et number 2
  - 1 sortie numérique : result

- Modification du nom, de la description et de l'icône



Exemple suite

- Créer les variables nécessaire dans init :

```
def __init__(self):
    GeneratedClass.__init__(self)
    self.number1 = None
    self.number2 = None
```

- Créer la méthode qui fait l'addition et envoie le résultat

```
def add(self):
    if (self.number1 != None and self.number2 != None):
        sum = self.number1 + self.number2
        # self.logger.info("sum : " + str(sum))
        self.result(sum)
        self.number1 = None
        self.number2 = None
```

- Traiter les entrées et appeler la méthode précédente

```
def onInput_number1(self, p):
    self.number1 = p
    self.logger.info("num1 : " + str(self.number1))
    self.add()
```

Exemple code final

Script editor

Add

```
class MyClass(GeneratedClass):
    def __init__(self):
        GeneratedClass.__init__(self)
        self.number1 = None
        self.number2 = None

    def onLoad(self):
        pass

    def onUnload(self):
        pass

    def onInput_number1(self, p):
        self.number1 = p
        self.logger.info("num1 : " + str(self.number1))
        self.add()

    def onInput_number2(self, p):
        self.number2 = p
        self.logger.info("num2 : " + str(self.number2))
        self.add()

    def add(self):
        if (self.number1 != None and self.number2 != None):
            sum = self.number1 + self.number2
            self.logger.info("sum : " + str(sum))
            self.result(sum)
            self.number1 = None
            self.number2 = None
```

Application 1 : librairie mathématique

- Certaines fonctions peuvent être utiles, et ne sont pas présentes au niveau des boîtes.

- Prévoir les boîtes suivantes (et les enregistrer dans la librairie perso) :
  - Conversion string => int
  - Conversion int => string
  - Soustraction de deux nombres
  - Elévation au carré
  - Racine d'un nombre

Utilisation des API

- Toutes les fonctions avancées de Nao sont accessibles
- Doc complète :
- http://doc.aldebaran.com/2-1/naoqi/index.html

- Utiliser des API :
  - Déclarer un proxy vers le module voulu (dans init ou onLoad) :

self.XXProxy = ALProxy("Nom module")

  - Utiliser les méthodes du proxy :

self.XXProxy.Methode(…)

Exemple : Hello World


```python
class MyClass(GeneratedClass):
    def __init__(self):
        GeneratedClass.__init__(self)
        self.tts = ALProxy("ALTextToSpeech")

    def onLoad(self):
        #put initialization code here
        pass

    def onUnload(self):
        #put clean-up code here
        pass

    def onInput_onStart(self):
        self.tts.say("Hello")
        self.onStopped() #activate the output of the box
        pass
```

Application 2 : Apprenez à compter

- Nao va proposer deux chiffres tirés au sort, et l'utilisateur devra indiquer la somme produite.

- Si elle est correcte, Nao va féliciter le joueur, sinon il donne la bonne réponse.

- Le jeu continue tant que le joueur ne demande pas de s'arrêter.