VERSION 1.0

AriGato

NAO Documentation: User Manual

Human-Robot Interaction

CREATED BY YOUR NAME

ARIGATO, CENTRAL WASHINGTON UNIVERSITY
ELLENSBURG, WA

"Our mission is to create and showcase meaningful and exciting human-to-robot interaction using the Aldebaran NAO robots recently required by CWU." – AriGato Robotics

CONTENTS

SECTION 1: INTRODUCTION TO THE NAO ROBOT

1.1 WHAT IS NAO?

NAO is an autonomous, humanoid, fully programmable robot. NAO robots are capable of 25 degrees of freedom, and thanks to their humanoid nature and design, are able to walk around, adapt, and interact with their surrounding environment. Furthermore, NAO has 4 directional microphones, loudspeakers, and 2 cameras capable of filming and analyzing the robot's environment, and human faces, for example. NAO is additionally capable of connecting to the internet by means of ethernet or Wi-Fi – this enables features such as http requests or big data analytics using the cloud.

All of these features add up to NAO's capabilities essentially being limitless, it is truly up to the developer's imagination to decide what the robot will eventually be capable of. Development for NAO can primarily be conducted in either Python or C++, though some other programming languages have small amounts of support as well (e.g. Java, MatLab).

Additional Specifications:

- Dimensions: 22.6 x 10.8 x 12.2 inches (574 x 311 x 275 mm)
- Weight: 12.08 pounds (5.48 kg)
- Autonomous Battery Life: 60 minutes active use, 90 minutes stationary use
- Operating System: Linux-Based NAOqi 2.8 (Linux Distro: Gentoo)
- Processor: Intel Atom E3845 @ 1.91 GHz

1.2 WHO MADE NAO?

The initial development of the NAO robot began as early as 2004. NAO was created by a French company known as Aldebaran, who was later acquired by SoftBank Robotics, a company based out of Japan, in 2015. The first public version of the NAO robot was released in 2008, however the version this project will be focusing on (NAO v6, or NAO Next Gen) was released to the public in 2014. Aldebaran also created the "Choregraphe" software that a bulk of the development of custom modules for NAO are made in.

1.3 DEFINITIONS, ACRONYMS, & ABBREVIATIONS

ALDEBARAN

A French robotics company, acquired by SoftBank Robotics in 2015. Developed NAO and Choregraphe.

API

Application Program Interface.

AUTONOMOUS LIFE

The application housed on NAO that "gives it life". With Autonomous Life activated, NAO becomes visually alive – by moving, "breathing", and being aware of its surroundings.

CHOREGRAPHE
A multi-platform desktop application that allows users to create animations and behaviors for the NAO unit, and test them in both simulated and real environments. It also allows users to monitor the NAO's visual and audio sensors as well as communication and error logs.

ETHERNET
A common form of network cable. It allows a connected device to join a local area network (LAN) in order to connect to and browse the internet.

LIBRARY
A collection of well-defined resources and implementations of behavior, written for/in a particular programming language for use by other developers to simplify and speed up development for a system.

MODULE
A program that is developed for use with/on the NAO robot. A module is classified as a feature for NAO that has a "trigger phrase", and then a timeline of events using Python code to have the NAO move around or speak certain phrases in responds to the corresponding trigger phrase.

NAO
An autonomous, fully programmable, humanoid robot designed by Aldebaran Robots.

NAOQI
A Linux-based operating system stored in the robot's memory at all times; used for running and controlling features, programs, and modules.

SEE
The NAO robots cannot "see" in a physical sense but has cameras that it can use to record images to analyze and identify its surroundings.

SENSOR
Measures the robot's configurations, conditions, and its environment and sends such information to the robot for processing.

SOFTWARE
A set of computer instructions used to obtain input, and then manipulate that input in order to generate relevant output in terms of function and performance as specified by the user.

TRIGGER PHRASE
A phrase that corresponds to a specific module housed on the NAO robot. When NAO hears a trigger phrase, the module it corresponds to will begin.

USER
A person who will interact with and make use of the NAO robot's various capabilities.

WI-FI

Short for “Wireless Fidelity”. A means of allowing computers, smartphones, and other internet-enabled devices to communicate with one another wirelessly.

1.4 REFERENCES

[1] NAO Lab Documentation/Info: https://team.inria.fr/perception/demos/naolab/
[2] NAO, NAOqi, Choregraph Documentation: http://doc.aldebaran.com/
[3] NAO, Technical Guide: http://doc.aldebaran.com/2-1/family/index.html
[4] Engineering NYU, Intro to Robotics: http://engineering.nyu.edu/mechatronics/smart/pdf/Intro2Robotics.pdf
[5] SoftBank Robotics Community Forums: https://community.ald.softbankrobotics.com/

SECTION 2: NAO’S CAPABILITIES

2.1 INITIAL (PRE-BUILT) CAPABILITIES

The original capabilities of the NAO are put on the robot by subscribing to the Aldebaran “Basic Channel.” Official documentation on this channel lists and describes its capabilities.[1] Below is a small table listing some of the pre-built phrases that NAO can respond to, most of which are available in English, French, and Japanese Language settings.

“How are you?”	“Can you say goodbye?”	“What can you do?”
“Tell me all you can do.”	“How do I install an application?”	“How do I start an application?”
“What did I say?”	“Can you repeat please?”	“What is your IP address?”
“Are you connected to Internet?”	“What languages so you speak?	“Speak French.”
“Can you speak French?”	“Speak Japanese.”	“Speak Chinese”
“Can you speak Chinese”	“Speak softer”	“Speak louder”
“Can you stand up?”	“Can you sit down?”	“Crouch.”
“Lay down”	“Lift your right arm”	“Lay down on your back”
“Lay down on your belly.”	“Stop Looking at me”	“What is your name?”
“Introduce yourself”	“How much do you weigh?”	“How tall are you?”

[1] http://doc.aldebaran.com/2-1/nao/basic_channel_conversation.html

2.2 VERBAL RESPONSES (Q&A)

CAN YOU DO MY HOMEWORK?

Verbal queues:

- “Can you do my homework?”
- “Can you do my work?”
- “Can you do my math homework?”
- “Can you do my Computer Science homework?”
- Can you do my CS homework?”
- “Can you code for me?”
- “Can you do my lab for me?”

Description: NAO will respond by telling the user that they should do their own work!

DEVELOPERS

Verbal queues:

- “Who is developing your programs?”
- “Who is working on you?”
- “Who is in the capstone project?”

Description: NAO will list the team members of the AriGato capstone project.

FAVORITE CLASS

Verbal queues:

- “What is your favorite class?”
- “What is your favorite C.S. class?”
- “What is your favorite class at C.W.U.?”

Description: NAO will respond with his favorite computer science class.

FAVORITE COLOR

Verbal queues:

- “Do you like any colors?”
- “Do you have a favorite color?”
- “What is your favorite color?”

Description: NAO will respond with his favorite color.

FAVORITE PROFESSOR

Verbal queues:

- “Who is your favorite professor?”
- “Do you know any professors?”

Description: NAO will respond stating that his favorite professor is the AriGato group's amazing supervisor, Dr. Davendra!!

FAVORITE SONG

Verbal queues:

- "Do you have a song you like?"
- "Do you like any songs?"
- "Do you like music?"
- "Do you have a favorite song?"
- "What is your favorite song?"

Description: NAO will respond with his favorite song, "Single Ladies"!

HOW OLD ARE YOU

Verbal queues:

- "How old are you?"
- "What is your age?"

Description: NAO will respond with a randomly selected humorous verbal response.

JOKES

Verbal queues:

- "Do you know any jokes?"
- "Tell me a joke."
- "Can you tell me any jokes?"

Description: NAO will respond with a randomly selected interactive joke.

SEE YOU LATER, ALLIGATOR

Verbal queues:

- "See you later Alligator"

Description: NAO will respond with "in a while, crocodile".

SING THE ANTHEM

Verbal queues:

- "Sing the [national] anthem."
- "Sing [national] anthem."
- "Can you sing the [national] anthem?"
- "Do you know the [national] anthem?"
- "Sing the Star Spangled Banner."

Description: NAO will begin to “sing” an auto-tuned version of the American National Anthem (an .mp3 file is played over its loudspeakers) and patriotically place its hand over its heart.

Note: “[national]” in the Verbal Queues section means the word is optional within the command.

WHAT TIME IS IT?

Verbal queues:

- “What time is it?”
- “Do you know the time?”
- “Can you tell me the time?”
- “Tell me the time”

Description: NAO will respond with the current Time according to what time zone is given by IP. If Nao is connected to a VPN the time may not be local time.

2.3 INTERNET-BASED REQUESTS & RESPONSES

ROBOT MOVIE INFORMATION

Verbal queues:

- “Tell me about a movie.”
- “Do you know any movies?
- “Do you like movies?”

Description: NAO uses an HTTP request to get information about various robot movies (from a finite list seen below) from the RottenTomatoes.com API, and then repeats the information gathered from the website to the user.

Complete Movie List:

- Astro Boy
- Big Hero 6
- Blade Runner
- Chappie
- I Robot
- The Iron Giant
- Making Mr. Right
- The Matrix
- Pacific Rim
- Real Steel
- Robocop
- Robot & Frank

- Robot Overlords
- Robots
- Saturn 3
- Spare Parts
- Surrogates
- The Terminator
- Transformers
- Wall-E

TEMPERATURE

Verbal queues:

- “What is the current temperature of Ellensburg?”
- “What’s the current temperature?”
- “What is the temperature?”
- “How hot is it outside?”
- “How cold is it outside?”
- “How hot is it?”
- “How cold is it?”
- “What about the weather?”

Description: NAO retrieves weather information from OpenWeatherMap.com and replies with the current temperature* in Ellensburg Washington. Due to this module being intended for use in Ellensburg exclusively, if one wants to change the city, they will have to go into the module’s code and change it manually.

*This module works only in Fahrenheit, not Celsius.

WEATHER-BASED CLOTHING RECOMMENDER

Verbal queues:

- “What should I wear outside today?”
- “What should I wear outside?”
- “Should I wear a jacket today?”
- “Should I wear a jacket?”
- “What should I wear today?”
- “What should I wear?”
- “Do I need a jacket today?”
- “Do I need a jacket?”
- “What clothing should I wear today?”
- “What clothing should I wear?”
- “What outfit should I wear today?”
- “What outfit should I wear?”

Description: NAO retrieves weather information from OpenWeatherMap.com and replies with a recommendation for types of clothing to wear outdoors based upon the current weather conditions (e.g., windy weather would cause NAO to recommend a jacket). Due to this module being intended for use in Ellensburg exclusively, if one wants to change the city, they will have to go into the module's code and change it manually.

2.4 FACIAL DETECTION & RECOGNITION RESPONSES

AGE GUESSER

Verbal queues:

- "Can you guess how old I am?"
- "Guess my age."
- "How old do you think I am?"
- "What is my age?"
- "How old am I?"

Description: NAO will use its facial detection and mapping abilities to attempt a guess at the user's age. The guesser is not the most accurate, however the module is enjoyable and humorous.

FACIAL RECOGNITION

Verbal queues:

- "Do you know me?"
- "Do you remember me?"
- "Do you know my name?"
- "Do you know who I am?"

Description: Provided that a human face is within view - if NAO can recognize the face within 6 seconds, NAO will greet the person, if not, the module will time out, and NAO will say "Sorry, I do not recognize you".

Note: Learning new faces is currently only possible when the NAO unit is connected to a computer and the "Learn Face" box within Choregraphe is ran (i.e., there is no type of "Learn my face" command to have NAO learn a face on the fly).

MOOD GUESSER

Verbal queues:

- "Can you guess how I'm feeling?"
- "Guess my mood."
- "How do you think I'm feeling?"
- "What is my mood?"

Description: NAO will use its facial detection and mapping abilities to attempt a guess at the user's current mood based upon their facial expression. Similar to the "Age Guesser" module, the guesser is not the most accurate, however the module is enjoyable and humorous.

The moods that NAO recognizes are:

- **Happy** (NAO recognizes a wide smile).
- **Surprised** (NAO Recognizes a shocked looking face, categorized by a wide, open mouth).
- **Sad** (NAO recognizes a frown)
- **Angry** (NAO recognizes a scowl, categorized by a pursed frown, middle-lowered and furrowed eyebrows, flared nostrils).
- **Neutral** (NAO recognizes a straight, bored-looking face with no visible emotion being shown).

2.5 BASIC MOVEMENT OPTIONS

Jazz Hands

Verbal queues:

- "Jazz hands."
- "Do jazz hands."
- "Be jazzy."
- "Can you do jazz hands?"

Description: NAO performs "jazz hands" with its hands, essentially a small wrist motion.

Move Fingers

Verbal queues:

- "NAO, wiggle your fingers."
- "Move fingers."
- "Can you wiggle your fingers."
- "Do your fingers move."
- "Move your fingers."

Description: NAO will open and close its hands to demonstrate the mobility of its hands and fingers.

Nod Yes

Verbal queues:

- "Nod yes."
- "Can you nod for me?"
- "Nod your head."
- "Nod your head for me."

Description: NAO will move its head up and down to demonstrate the mobility of its head/neck.

RAISE LEFT/RIGHT FOOT

Verbal queues:

- “Raise your left/right foot.”
- “Raise your left/right leg.”
- “Move your left/right foot.”
- “Move your left/right leg.”
- “Balance on your left/right foot.”

Description: NAO will lean to the side and begin to balance on one leg while lifting the corresponding foot in order to demonstrate the mobility of its legs and potential balancing capabilities.

TURN AROUND

Verbal queues:

- “Turn around”
- “Spin around”
- “Can you turn around?”

Description: Nao will turn 180 degrees

TURN HEAD LEFT/RIGHT

Verbal queues:

- “Turn your head left/right.”
- “Turn head left/right.”
- “Head left/right.”
- “Move your head to the left/right.”
- “Look left/right.”

Description: NAO will turn its head to the corresponding direction in order to further demonstrate the mobility of its head/neck.

TURN LEFT/RIGHT

Verbal queues:

- “Turn left/right”
- “Rotate left/right”
- “Can you turn left/right?”
- “Can you rotate left/right?”

Description: Nao will ask the desired degrees to turn, and then turn in the direction and distance accordingly.

WALK FORWARD/BACKWARD/LEFT/RIGHT

Verbal queues:

- “Walk <direction>.”
- “Move <direction>.”
- “Step <direction>.”
- “Take a step <direction>.”
- “Move your legs for me”
- “Move your legs”

Description: NAO will move 0.2 meters in the desired direction. The last two queues will make Nao walk forward.

Note: Directional options include forward, backward, left, & right.

2.6 ADVANCED MOVEMENT OPTIONS

DANCE MOVES

Verbal queues:

- “Can you dance?”
- “Can you dance like Beyoncé?”
- “Can you dance to Single Ladies?”
- “Do the Beyoncé.”

Description: NAO will do a short dance routine, coordinated to Beyoncé’s “Single Ladies”, which will play over its loudspeakers during the dance.

PUSHUPS

Verbal queues:

- “Pushups.”
- “Do pushups.”
- “Do some pushups.”
- “Can you do pushups?”
- “Do you know how to do pushups?”
- “Can you do press-ups?”

Description: NAO will ask how many pushups you would like to be performed (he can do between 1 and 10 inclusive), and will proceed to do that many pushups.

WARNING

NAO **must** have plenty of clear and empty space around, in front, and behind it to do pushups, and must be kept away from any ledges - otherwise damage to the NAO unit is highly possible. Furthermore, making NAO do excessive amounts of pushups without a break can lead to overheating motors and will cause NAO to shut down.

****

2.7 MISCELLANEOUS MODULES

NO SWEARING

Description: NAO listens for swear words or other foul language, and responds to the user telling them that they should not use such language.

Note: This is a persistent module, meaning NAO is running it at all times. There is no specific command to trigger this module, instead NAO listens for words from a list of "bad words" to trigger it.

STOP MODULE

Verbal queues:

- "Stop."
- "Exit."

Description: NAO immediately stops the module it is currently running.

Note: This feature is available on modules that might take a while to complete, or involve multiple responses. It is not available on every module.

SECTION 3: CREATING YOUR OWN NAO MODULES

3.1 INTRODUCTION TO CHOREGRAPHE

Choregraphe is multi-platform desktop application that allows users to visually program modules for the NAO robot (as well as other robot models created by SoftBank Robotics, such as Pepper). Choregraphe links up with the NAOqi framework (discussed in the following subsection), enabling users to easily create animations and behaviors for NAO, as well as test their creations using a simulated virtual NAO robot, or a physical robot connected via ethernet.

The Choregraphe software comes packaged with a ton of pre-built functions for NAO (known as "boxes" within Choregraphe) such as a "Say Box" which easily allows for text-to-speech programs, or a "Movement Box" which users can take advantage of to have their NAO robot move a specific distance in a specific direction. The biggest draw of Choregraphe however is the ability it gives users to create their own custom boxes, which can be combined to create an entirely custom sequence of events (known as a module) for NAO. These custom boxes can be written in a multitude of programming languages, but this documentation will be focusing on the Python language, as that is what AriGato has solely used in their project.

3.2 THE NAOQI FRAMEWORK

"NAOqi" is the NAO robot's operating system. It is a programming framework that is built and runs off of the Gentoo Linux Distribution. NAOqi is the main software residing in NAO's memory unit, and controls all of the robot's motors, sensors, and functionalities. According to Aldebaran's documentation, "it answers to common robotics needs including: parallelism, resources, synchronization, [and] events[2]".

NAOqi offers a fully fleshed-out API for both C++ and Python, giving users access and manipulation of the full range of NAO's capabilities. There is a vast library of classes and their respective functions that can be called and expanded upon to create your own modules. A few examples of these classes are "ALTextToSpeech", "ALMotion", and "ALLeds", which each offer a variety of functions related to the class (e.g., ALMotion has a "moveTo" function that enables the NAO to walk). The "boxes" discussed in Section 3.1 rely on these API classes and functions to easily create modules for NAO.

3.3 CREATING YOUR FIRST MODULE

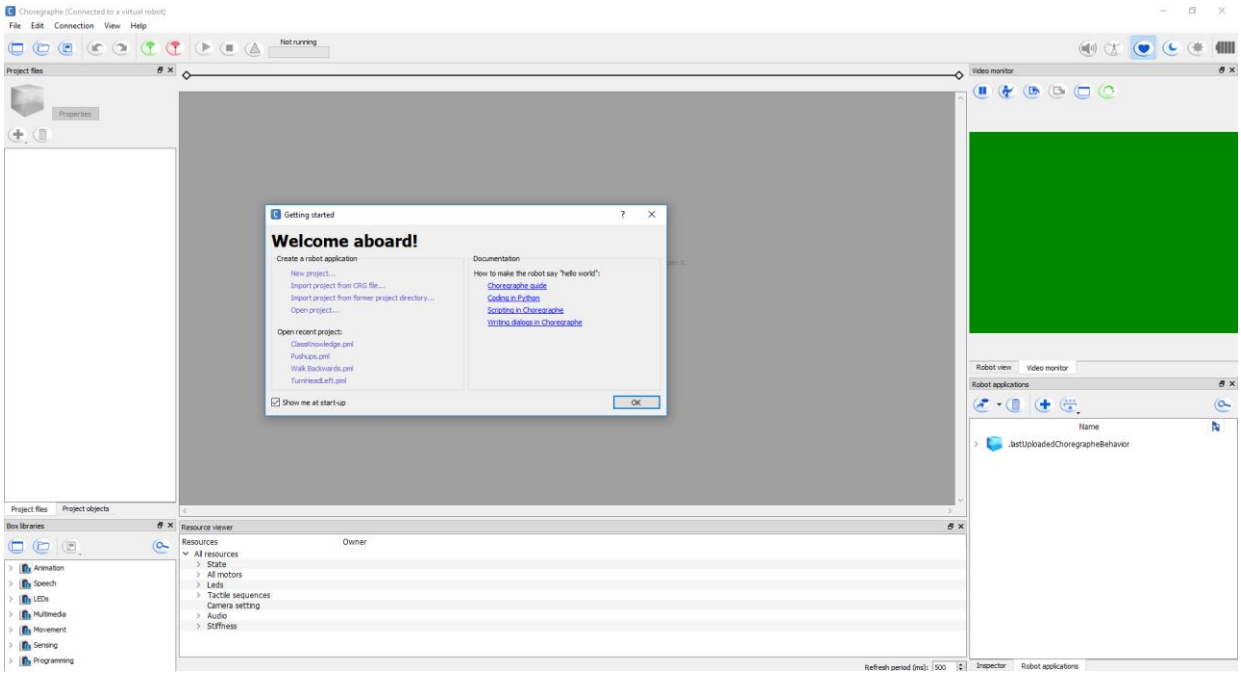


FIGURE 1 - FIRST OPENING CHOREGRAPHE

When first opening Choregraphe a window will pop up saying "Welcome aboard!" It will have a documentation section with a quick "Hello World" tutorial, and links to recent projects if there are any. To begin, either click "New project…" or exit out of the window, which will create a new project anyway. Start by navigating to the "File" menu in the top left corner of Choregraphe, and selecting "Project Properties".

[2] NAOqi Documentation Page: http://doc.aldebaran.com/1-14/dev/naoqi/index.html

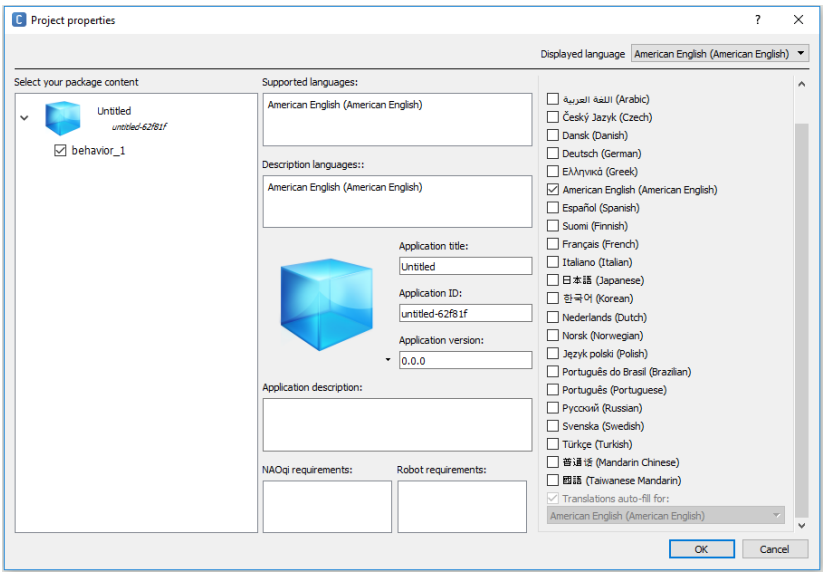


FIGURE 2 - PROJECT PROPERTIES

Fill in the "Application title" section with an accurate and descriptive title, so that when you eventually install your module onto the robot you will know exactly which module it is (it will otherwise be untitled which can become quickly confusing). Next click on the text "behavior_1" under the blue cube in the "Select your package content" section (upper left) of the window shown above in figure 2.

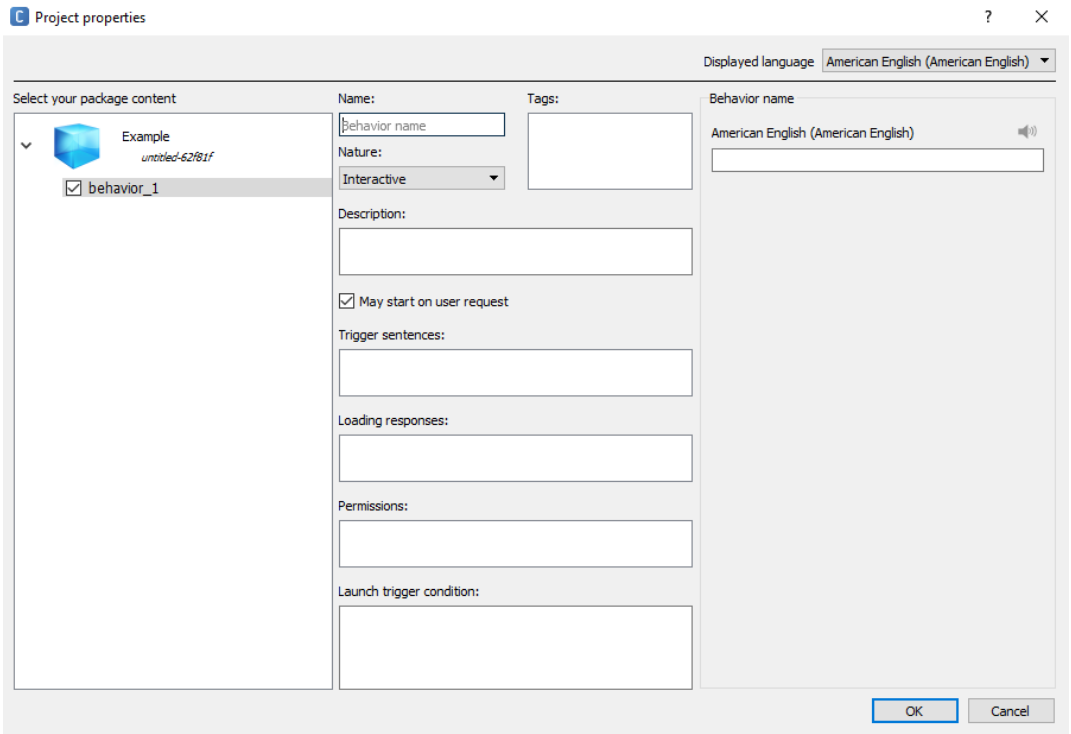


FIGURE 3 - BEHAVIOR PROPERTIES

Fill out the following sections shown in figure 3 after clicking “behavior_1”:

- **Name:** Make sure your module’s name is self-documenting (i.e., The name should be relevant to what the module does). A good example for a simple “Hello World” module would be “hello”, and a bad example would be “module 1”, or “myModule”.
- **Description**: This section will let future users know what your module is supposed to do, and possibly why you have created the module (to serve some certain purpose, for example). It is best to have your description be straight to the point, and not too long – but also informative with all the relevant information for the module included.
- **Nature**: Nature has three possible settings: “Interactive”, “Solitary”, and “No Nature”. **Make sure your module’s nature is set to Interactive**. This means that a user can trigger the module with verbal trigger sentences while NAO’s autonomous life feature is on. Solitary means that NAO will perform the module when he is not being interacted with, and can be interrupted at any time with Interactive modules. There is no available description for No Nature, so it is best not to make use of this mode.
- **Trigger sentences:** Trigger sentences are what NAO listens for to perform a corresponding module. Make sure each trigger sentence is unique, such that they don’t share a phrase with another module that is already on the robot. It is also good to make a couple of similar trigger phrases for each module, so that the module can be more generally commanded.
  (Example: For an arm raising module, instead of just saying “lift arm”, also put “lift YOUR arm”, “Can your lift your arm” and “lift your arm please.”)
- **Loading responses:** Loading responses are intended to be what the NAO unit will say after being told a trigger phrase, before the module is actually performed. However, in AriGato’s experience, these loading responses do not actually work, and the robot only ever uses the default phrases of “Okay” and “Let’s go!”.
- **Permissions:** Permissions gives NAO the ability to perform a module while in the process of sitting down, standing up, or sitting in the charging station. The charging station is for NAO models that are not currently in the possession of CWU, thus being not relevant to the AriGato project or this documentation.
- **Launch trigger condition:** Launch trigger conditions gives NAO the ability to perform the module autonomously (i.e., without an user prompting, according to the conditions described). **DO NOT** use trigger conditions on modules that you wish only to be triggered by a user interaction.

Now that you have set all of the relevant project properties, navigate back to the main screen (known as the “workbench”) of Choregraphe. **Note:** For this tutorial we will be showing you have to create custom boxes using Python code, but will also make short mention of how to find and use the pre-built functions that are packaged with Choregraphe.

Right click anywhere within the empty workbench, and hover your mouse over the “Create a new box” option, inside of the submenu that appears, select “Python…”. Your screen should open a new dialog box called “Edit box”, and look the same as below in Figure 4:

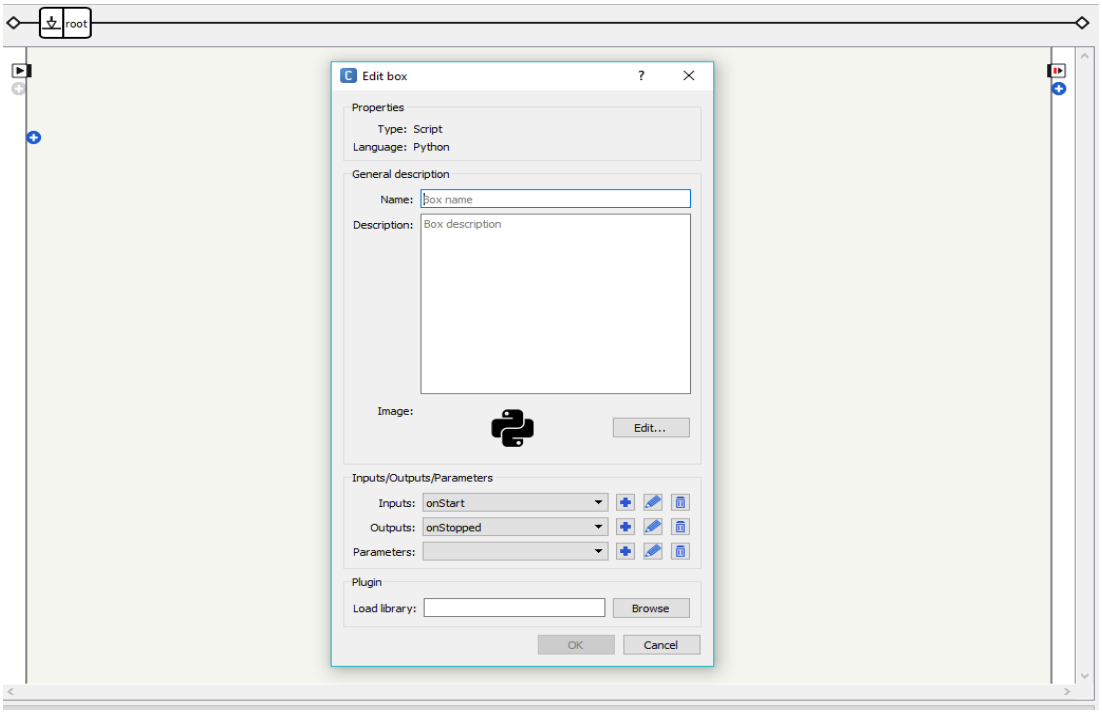
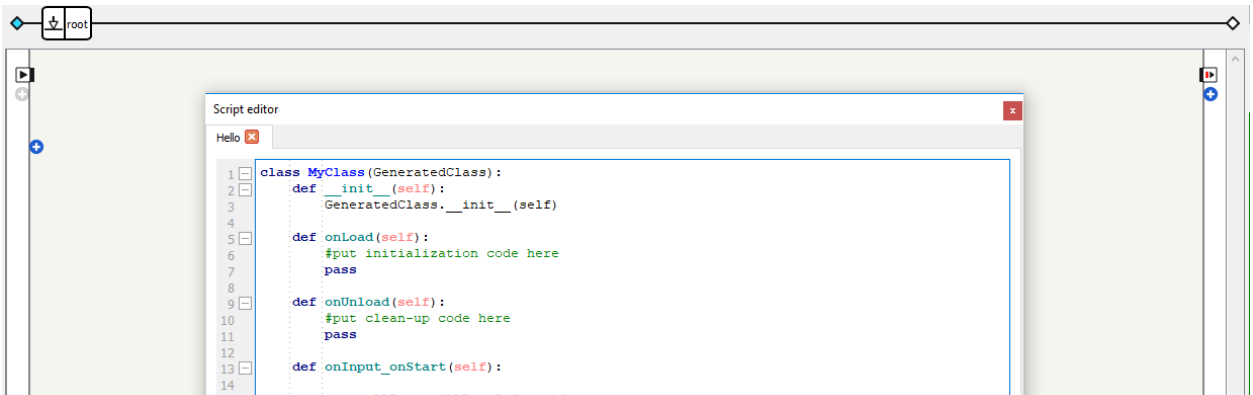


FIGURE 4 - EDIT BOX

Fill out the following sections shown in figure 4 after clicking "Python…":

- **Name:** Your box's name should follow suit with the module and application names you've previously set, in other words, it should be self-documenting and accurate to what the box will be doing. Think of this like a function name while programming. As previously stated, a good box name for a simple Hello World code would be something like "TextToSpeechBox", or "SayHelloBox".
- **Description**: Similar to the project description that you've already written, this section should be a relatively short, but informative definition of what it is that your newly created box will be doing.
- **Input**: Allows the user to add multiple start inputs.
- **Output**: Allows different outputs to be added to the module
- **Parameters**: Allows parameters to be added to the module that can be used in the module.
- **Plugin**: If your module is dependent on your own libraries, you can add them in the load library section under plugin

To add the script into the module, double click your python box, and a python script window will open, as shown in figure 4. Your code will be written in the *onInput_onStart* function. Delete the "pass" from this function, and uncomment the "on_Stopped()"



The first line in out function in figure 5 (line 15), `tts = ALProxy("ALTextToSpeech")`, creates a proxy using the `ALProxy` function provided in NAO's API. This function takes a string as its parameter, describing what the proxy is for. "ALTextToSpeech" for instance, converts text to speech, and "ALMotion" has functions that gives you the ability to control Nao's motion.

The second line (line 16), `tts.say("Hello World!")`, uses the text to speech proxy we just made, calls its function `say()` using the String parameter "Hello World!" When this is called, this will make Nao say "Hello World!"

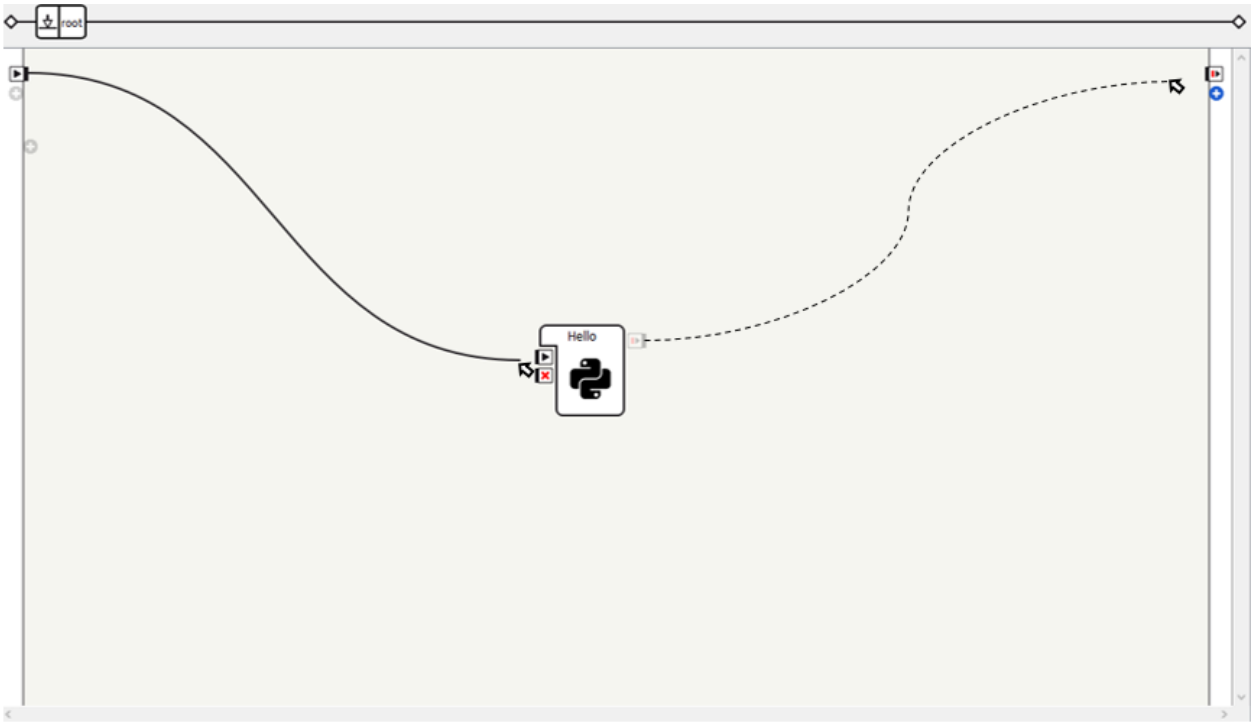


FIGURE 6 - DRAW THE LINES

To finish your project, create a connection from on start in the upper left corner to the on start on your python box as shown in figure 6. This starts your function when the behavior starts. If there is no connection from the on start in the upper left corner to any of your functions, your module will do nothing, and run infinitely. Drawing an arrow from the on stop on your function to the on stop in the upper right corner, as shown in figure 6, will exit the module. It is important to remember to connect something to the on stop in the upper right corner, so that there is some exit condition, however if the on start signal hits a dead end in your code it will just exit, so it is not completely necessary.

3.4 INTEGRATING YOUR MODULE ONTO NAO

To integrate your module first you must connect NAO to your computer via Ethernet cable. Then, hit the green button in the upper left corner, which will bring up a list of connections. Click your NAO robot, and then click the select button. You may have to wait for a moment while NAO connects to your computer.

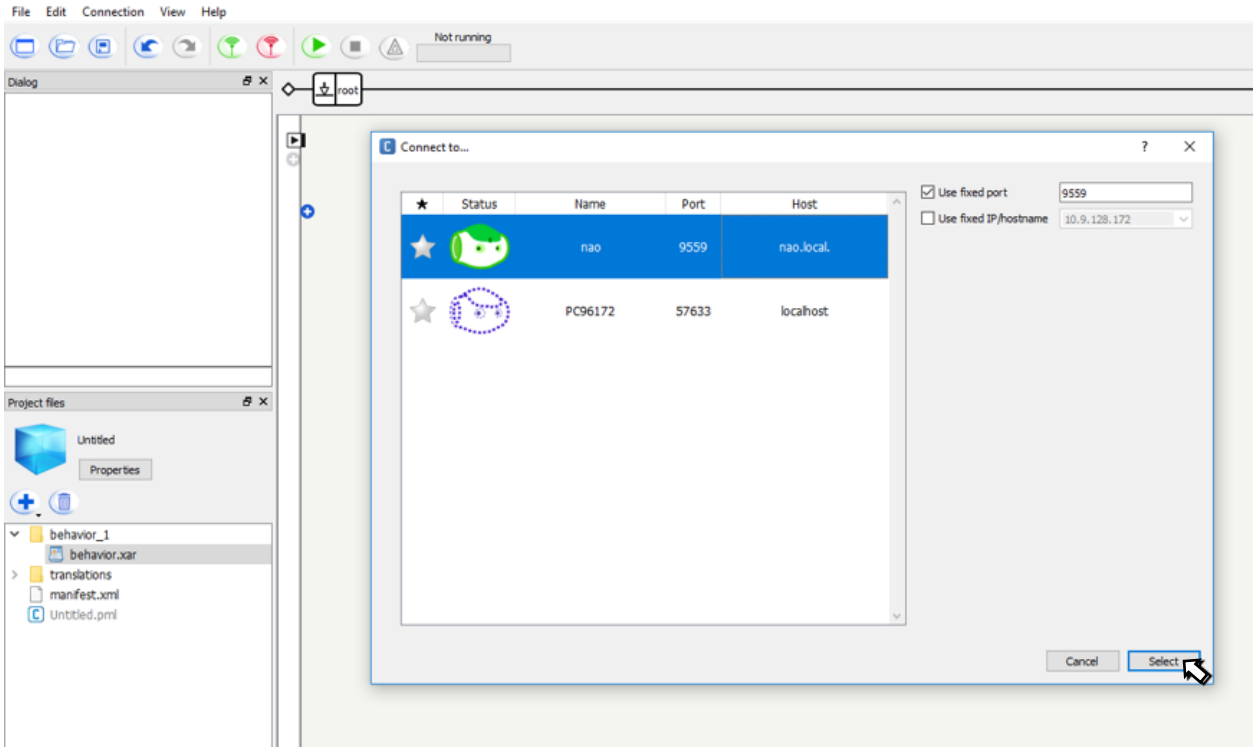


FIGURE 7 - CONNECTING YOUR ROBOT

Now adding your module should be simple. In the view tab, make sure "Robot Applications" is turned on. Then, click robot applications tab in the lower right corner, or wherever else it may appear, to open it. Now all you need to do is click the button with Nao's head on it in the upper right corner of the robot application box shown in figure 8, and the module will be loaded and ready to go.

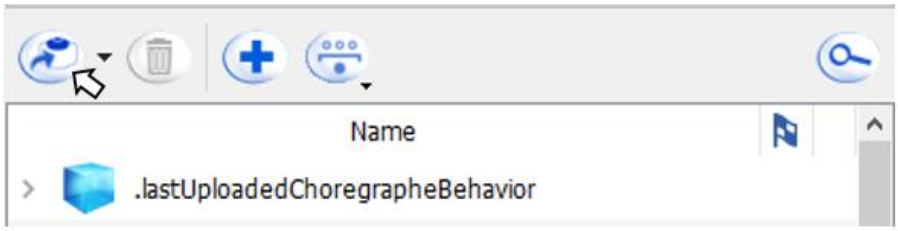


FIGURE 8 - ROBOT APPLICATIONS

SECTION 4: HELP

4.1 GENERAL FAQS

4.2 TROUBLESHOOTING