

Poppy

Documentation

release 1.1.0

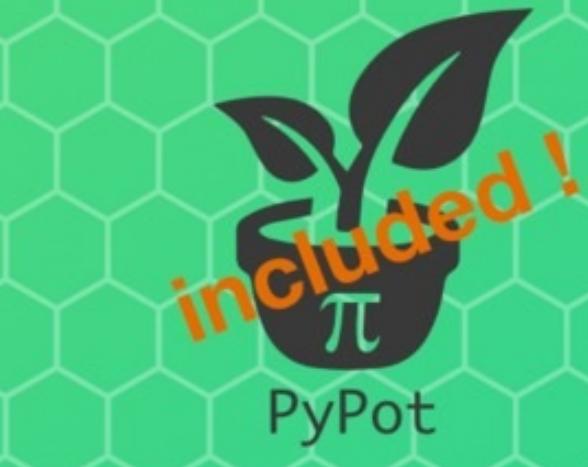


Table des matières

Introduction	1.1
Démarrer	1.2
Construire un robot	1.2.1
Se connecter au robot	1.2.2
Programmer le robot	1.2.3
Visualiser	1.2.4
Exemples de projets	1.2.5
Installation	1.3
Installer un client zeroconf	1.3.1
Démarrage d'un robot Poppy	1.3.2
Installer les logiciels Poppy	1.3.3
Installer le simulateur V-REP	1.3.4
Installer les pilotes USB vers port série	1.3.5
Installer une carte d'extension Poppy	1.3.6
Guides d'assemblage	2.1
Assembler le Ergo Jr	2.1.1
Assembler l'électronique	2.1.2
Configurer les moteurs	2.1.3
Construire la mécanique	2.1.4
Assembler le Poppy Humanoid	2.1.5
Assembler le Poppy Torso	2.1.6
Programmation	3.1
Programmer avec Snap!	3.1.1
Utiliser Jupyter notebooks	3.1.2
Programmer en Python	3.1.3
APIs Robot	3.1.4
Activités	3.2
Se connecter avec Snap4Arduino	3.2.1
De la simulation à un vrai robot	3.3
Snap! sur un vrai robot	3.3.1
Programmer avec Jupyter notebooks sur un vrai robot	3.3.2
Bibliothèques Logicielles	3.4
Pypot	3.4.1
Poppy-creature	3.4.2
Poppy Ergo Jr	3.4.3
Poppy Humanoid	3.4.4
Poppy Torso	3.4.5
Appendices	3.5
Réseau	3.5.1

Contribuer	3.5.2
FAQ	3.5.3

Introduction

À propos

Bienvenue dans l'espace de documentation de la plateforme robotique open source [Poppy](#).

La documentation est actuellement uniquement disponible en anglais mais est en cours de traduction. Nous vous invitons à la [consulter](#) et à nous contacter sur le [forum](#) si vous avez des questions.

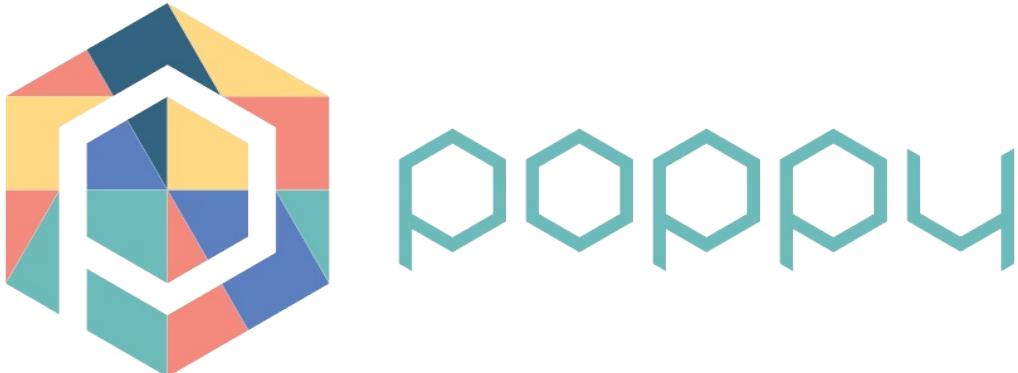
Cette documentation est sous licence Creative Commons (BY-SA). Les sources sont disponibles sous [GitHub](#) et nous vous invitons à participer à son amélioration continue en [remontant les bugs](#) et/ou [en participant](#) à son écriture !

Version

Dans ce document (version 1.0) vous trouverez les documentations pour:

- Poppy Humanoid 1.0.1 hardware
- Poppy Torso 1.0.1 hardware
- Poppy ErgoJr beta 6 hardware
- poppy_humanoid library version 1.1.1
- poppy_torso library version 1.1.5
- poppy_ergo_jr library version 1.4.0
- poppy.creatures library version 1.7.1
- pypot library version 2.10.0

Démarrer



Le [Projet Poppy](#) est un projet sous licence libre pour la création, l'utilisation et le partage de robots interactifs réalisés par impression 3D. Il rassemble une communauté inter-disciplinaire de débutants et d'experts, de scientifiques, d'enseignants, développeurs et artistes. Ils partagent une vision: les robots sont de puissants outils d'apprentissage, de création et de collaboration afin d'améliorer le projet. Ils développent de nouveaux comportements robotiques, créent des contenus pédagogiques, conçoivent des performances artistiques, améliorent le logiciel ou même créent de nouveaux robots.

La [Communauté Poppy](#) développe des créations robotiques faciles à construire, à personnaliser et à déployer. Nous promouvons les licences libres en partageant la matériel et le logiciel. Une plateforme Web associée permet à la communauté de partager ses expériences et de contribuer à son amélioration.

Pour faciliter ces échanges, deux supports sont disponibles :

- [Le forum poppy-project](#) pour obtenir de l'aide, discuter et échanger des idées.
- [GitHub](#) pour soumettre vos contributions.

Toutes les sources du projet Poppy (logicielles et matérielles) sont disponibles sur [GitHub](#).

Le projet Poppy a initialement été conçu au laboratoire [Inria Flowers](#).

Les créatures Poppy

Les créatures Poppy sont des robots sous licence libre, disponibles au téléchargement et pour être modifiés ([Licence Creative Commons avec attribution](#) pour le matériel et la licence [GPLv3](#) pour le logiciel). Elles ont été conçues avec ces mêmes principes directeurs.

Toutes les créatures Poppy:

- sont faites à partir de pièces imprimables en 3D et de moteurs Dynamixel,
- utilisent une carte d'extension pour le contrôle (un Raspberry Pi 2 ou Odroid pour les versions plus anciennes),
- sont basées sur une bibliothèque Python, [pypot](#), qui permet le contrôle des servomoteurs Dynamixel de manière simplifiée,
- ont une version de simulation disponible (basée sur [V-REP](#)),
- peuvent être contrôlées en utilisant un langage de programmation visuel ([Snap!](#) une variation de Scratch) et un langage textuel [Python](#). Elles sont aussi programmables par l'intermédiaire d'une API REST, ce qui permet de les contrôler à partir d'autre langages de programmation,
- viennent avec la documentation associée, des tutoriels, exemples et activités pédagogiques.

Elles peuvent être utilisées telles quelles, ou modifiées pour explorer de nouvelles formes, l'ajout de capteurs, etc...

Pour obtenir votre propre robot Poppy, vous pouvez soit :

- Vous procurer toutes les pièces vous-même en suivant la liste ci-dessous.
- Acheter un kit de robotique Poppy complet auprès de notre [détailleur officiel](#).

Poppy Ergo Jr

Le robot Poppy Ergo Jr est un petit bras robotique à cout modéré qui possède six degré de mouvements. Il est fait de 6 moteurs au prix abordable (des servos XL-320 Dynamixel) ainsi que de parties simples et imprimables en 3D.



Les pièces 3D ont été conçues pour être aisément imprimée sur un imprimante 3D courante. Les moteurs coûtent environ 20€ chacun. La carte électronique de contrôle est simple. Cela facilite la connexion de capteurs supplémentaires et est donc adapté à des objectifs pédagogiques.

Vous pouvez choisir parmi les trois outils pour équiper l'extrémité de son bras :

- Une lampe.
- Une pince.
- Un porte-stylo.

Les rivets employés rendent le montage et démontage des outils simples. Vous pouvez les adapter en fonction de l'activité choisie.

Le Poppy Ergo Jr est idéal pour débuter à manipuler des robots et apprendre la robotique sans difficultés. Il est simple à assembler, contrôler et d'un coût réduit.

Vous pouvez obtenir les pièces vous-mêmes en suivant la [liste des matériaux](#) (LDM) et imprimer les [pièces 3D](#) disponibles au format STL.

Pour plus d'informations, vérifiez le [guide d'assemblage de l'Ergo Jr](#).

L'Humanoïde Poppy

Il s'agit d'un robot humanoïde à 25 degrés de liberté possédant une colonne vertébrale complètement motorisée. Il est utilisé pour l'éducation, la recherche (marche, interactions entre humains et robots) ou les arts (dances, performances). Du simple bras à l'humanoïde complet, cette plateforme est activement utilisée dans les laboratoires, les écoles d'ingénieur, les FabLabs, et les projets artistiques.

Vous pouvez obtenir les pièces vous-mêmes en suivant la [liste des matériaux](#) (LDM) et imprimer les [pièces 3D](#) disponibles au formats STL, STEP et Solidworks 2014.



Le Torse Poppy

Il s'agit de la partie supérieure de l'Humanoïde Poppy (13 degrés de liberté). Le Torse Poppy est donc plus abordable que l'Humanoïde Poppy complet. Cela en fait une solution plus appropriée aux contextes éducatifs et associatifs. Le Torse Poppy peut être un bon média pour apprendre la science, la technologie, l'ingénierie et les mathématiques (STEM).

Vous pouvez obtenir les pièces vous-mêmes en suivant la [liste des matériaux](#). Les [modèles 3D](#) pour les pièces sont identiques à celles de l'Humanoïde Poppy, sans les jambes et avec un [support à ventouse supplémentaire](#).

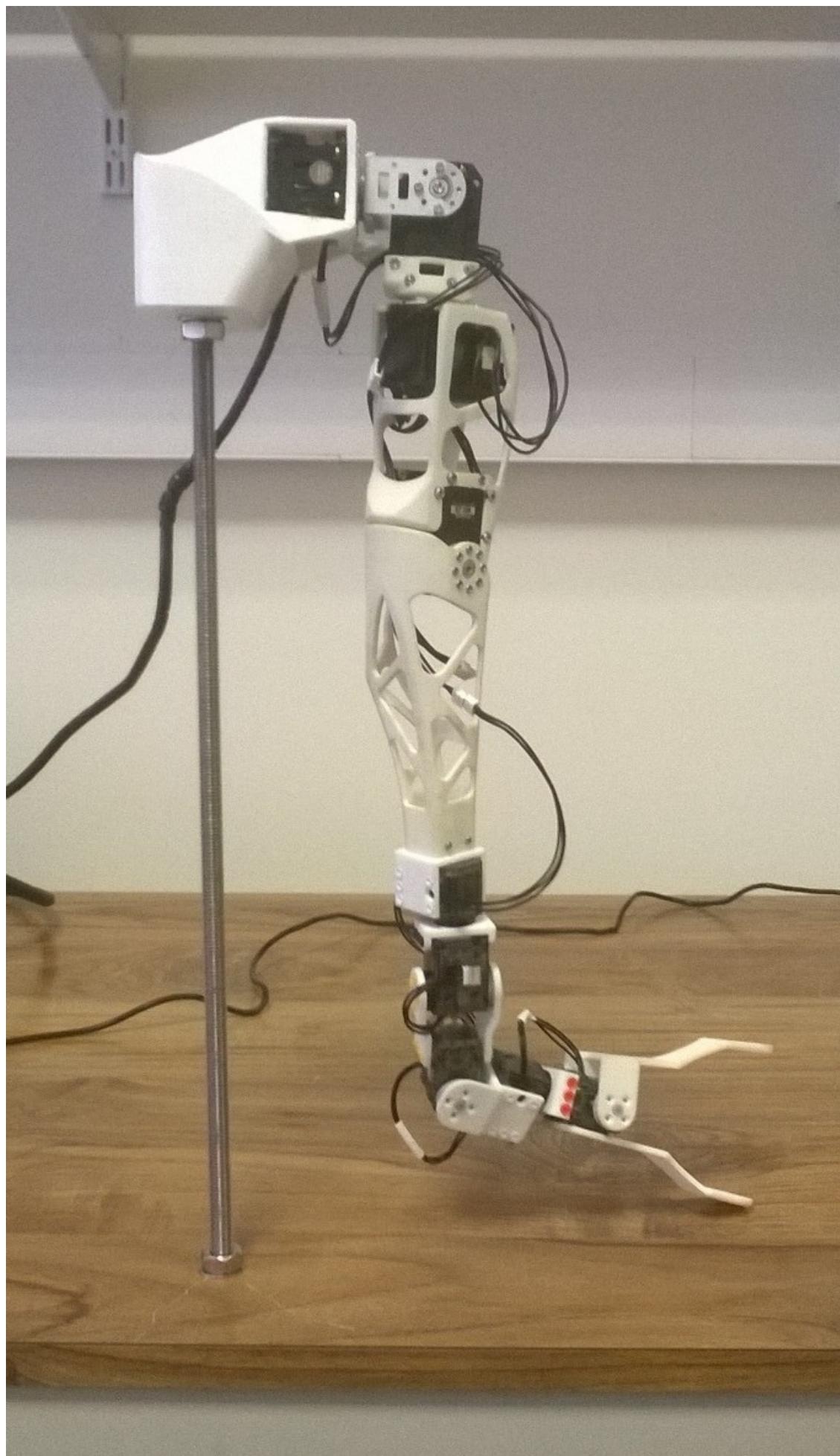


Autres Créatures Poppy

Un aspect clef du projet Poppy est de stimuler la créativité et l'expérimentation autour de la robotique. Nous essayons de fournir tous les outils requis pour la conception de nouveaux robots basés sur les mêmes briques technologiques. Plusieurs créatures sont en cours de développement au sein de la communauté. Certaines d'entre elles vous sont présentées ci-dessous.

Le Bras Droit Poppy (en construction)

Le Bras Droit Poppy est une créature basée sur le bras droit de l'Humanoïde Poppy, avec 3 moteurs XL-320 additionnels à son extrémité afin d'améliorer la portée et l'agilité du bras. Il utilise le même outil pince employé par l'Ergo Jr, conçu pour saisir des objets simples.



Le projet a été réalisé pendant le stage de [Joel Ortiz Sosa](#) au laboratoire Inria Flowers. Plus d'informations et les fichiers sources sont disponibles dans le [dépot Github correspondant](#).

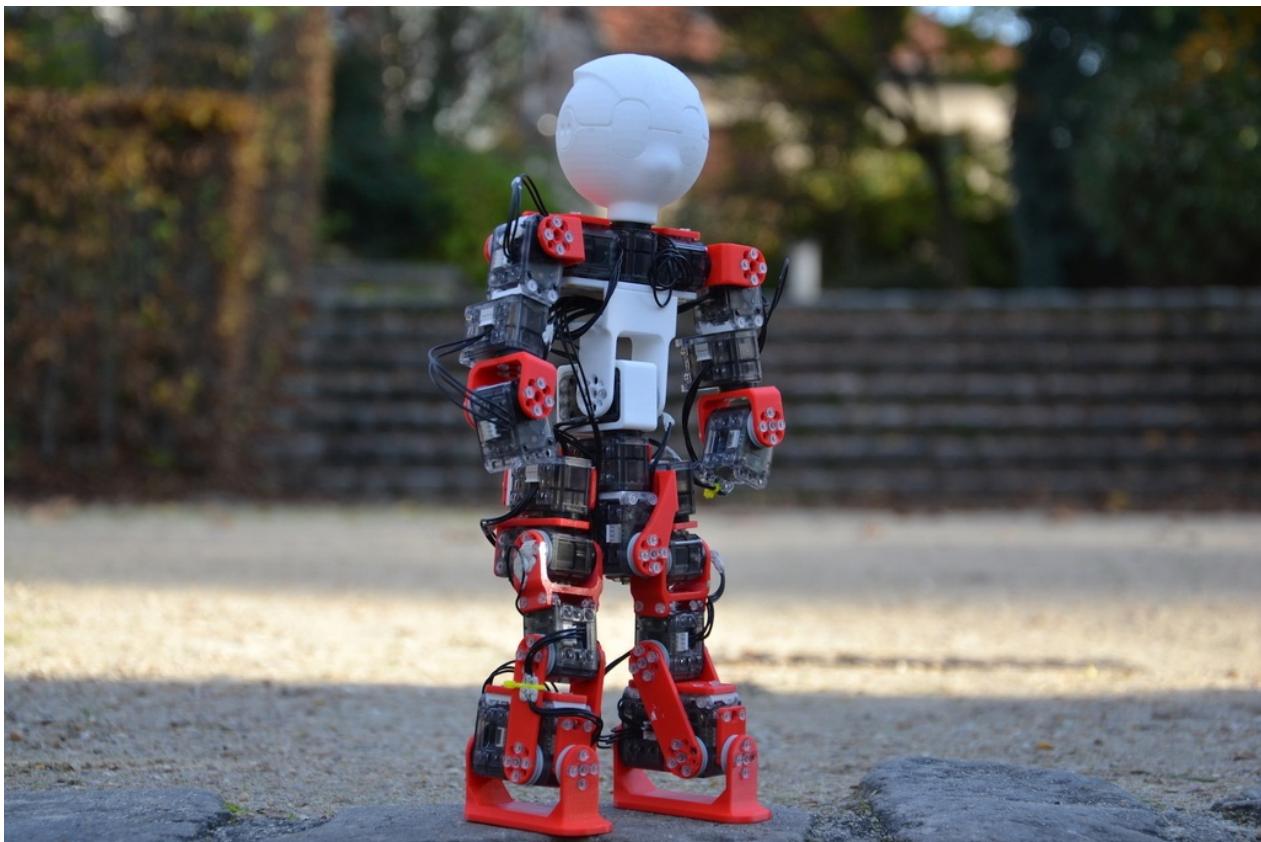
Humanoïdes plus petits et abordables

Heol

Heol - qui signifie "soleil" en Breton - est un humanoïde de 34cm de hauteur produit par l'association [Heol robotique](#). Il est composé de 23 moteurs, toutes les autres pièces sont imprimées en 3D. Il dépend aussi de la bibliothèque Pypot pour ses mouvements.

Le but d'Heol est de faire naître un sourire sur le visage des enfants. Il peut être un outil éducatif en devenant un support d'apprentissage de la programmation et de la conception mécanique.

Sa participation à la RoboCup (Tournoi international de football pour robots) est aussi envisagée.



[Poppyrate](#)

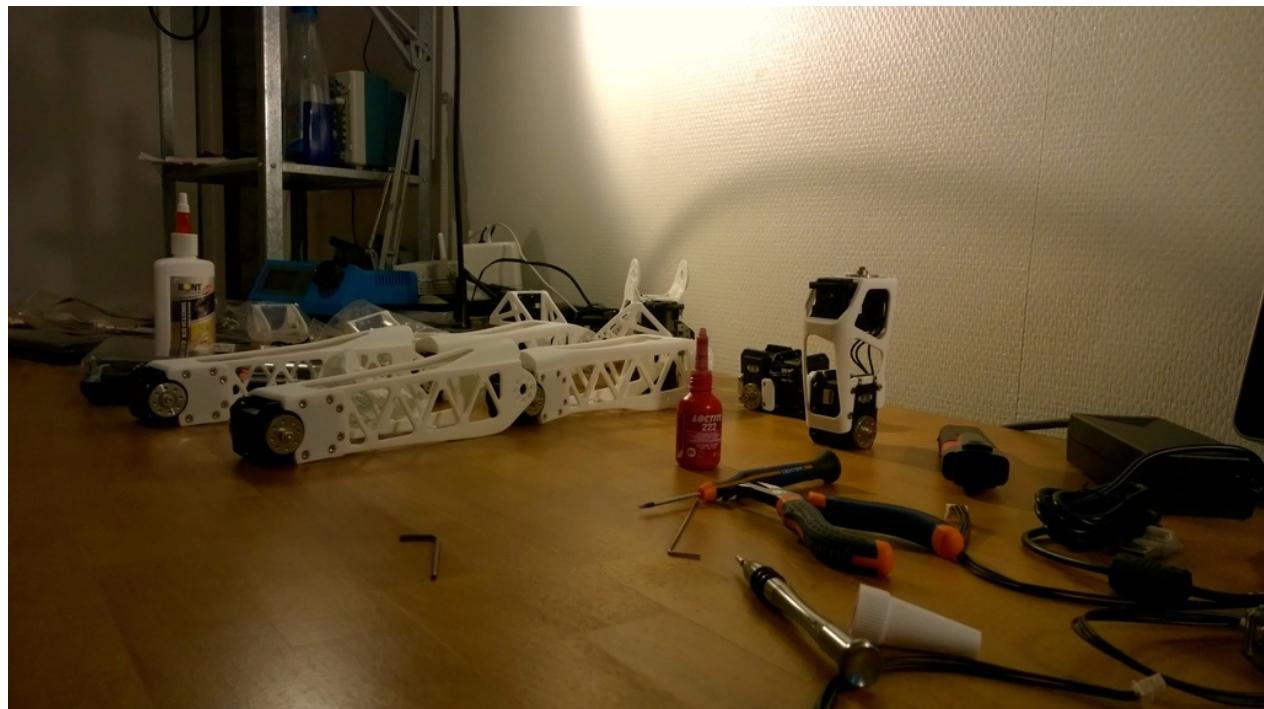
Il s'agit d'un robot basé sur l'Humanoïde Poppy. Il vise le développement d'une version moins chère grâce à sa taille réduite et l'usage de moteurs moins chers. La réduction de taille le rend aussi plus facile à imprimer sur une imprimante 3D courante. D'autres buts sont de le rendre aussi mobile et personnalisable que possible tant en conservant la compatibilité avec le logiciel Poppy.

Poppyrate peut être vendu en kit (avec ou sans les parties 3D imprimées). Il a été conçu par la société ZeCloud.



Pour plus d'informations, consultez le [site Web](#) - [Twitter](#) - [Facebook](#)!

Assembler le robot

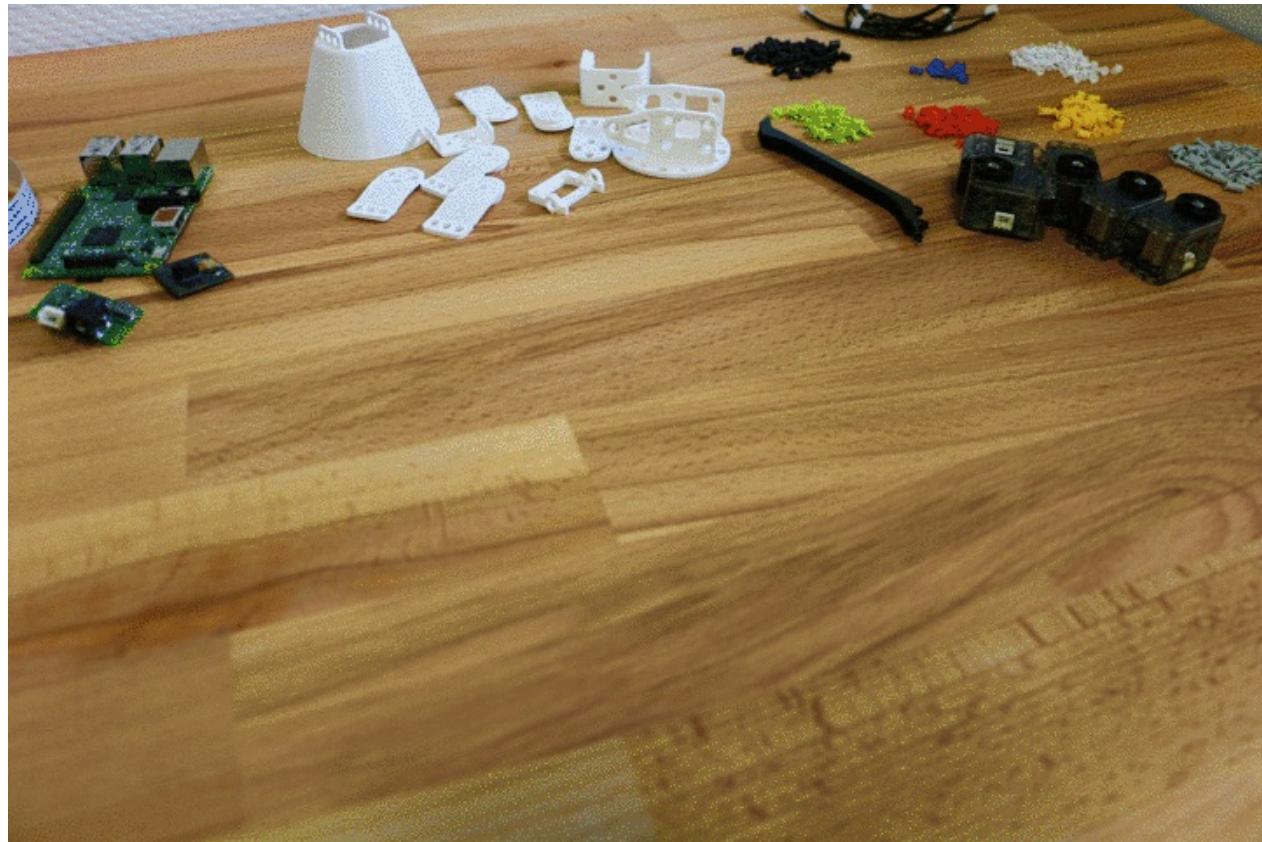


Suivant quel robot Poppy vous souhaitez utiliser, le temps d'assemblage, les compétences requises, les outils et la difficulté peuvent varier de beaucoup. Construire un Ergo Jr devrait prendre une heure et aucun outillage spécifique n'est requis, alors que l'assemblage de l'Humanoïde Poppy devrait prendre plusieurs jours et de nombreuses vis.

Cette section vise à vous donner des conseils et un aperçu de quelques points importants afin que vous en soyez familiers avant d'entamer la construction. Nous allons aussi vous diriger vers les chapitres dédiés où vous trouverez les ressources et les procédures d'assemblage détaillées pas-à-pas de chaque robot.

Assembler un Ergo Jr

Vous pouvez trouver la documentation d'assemblage complète dans le chapitre [assemblage pas-à-pas d'un Ergo Jr](#).



Le robot Ergo Jr a été conçu pour être un petit robot, bon marché et facile à utiliser. Les pièces 3D ont été faites pour être imprimées facilement sur une imprimante 3D standard et les moteurs (6 servos XL-320 Dynamixel) ne coûtent que 20\$ chacun.

Le Ergo Jr est très facile à construire et son appendice peut être facilement changé - vous pouvez choisir parmi plusieurs outils : une lampe, une pince, un support pour stylo...

Grâce aux rivets OLLO le robot est vraiment simple à assembler. Ces rivets peuvent être enlevés et ajoutés très facilement avec l'outil OLLO. Il ne devrait pas prendre plus d'une heure à être entièrement construit, ce qui permet une grande liberté de modification.

Mis à part le calibrage de l'orientation des moteurs, il n'y a pas de difficulté majeure. Si vous êtes familiers avec les briques Lego, vous devriez être capable d'assembler un Ergo Jr sans trop de problèmes! Les rivets sont pensés pour être faciles à assembler et désassembler, alors en cas de problème vous pouvez simplement redémarrer à zéro! Assurez vous de [configurer les moteurs](#) avant d'assembler le robot car il est plus difficile de le faire après!

Assembler un Torse ou un Humanoïde

Vous pouvez trouver la documentation d'assemblage complète dans le chapitre [assemblage pas-à-pas d'un Humanoïde Poppy](#).



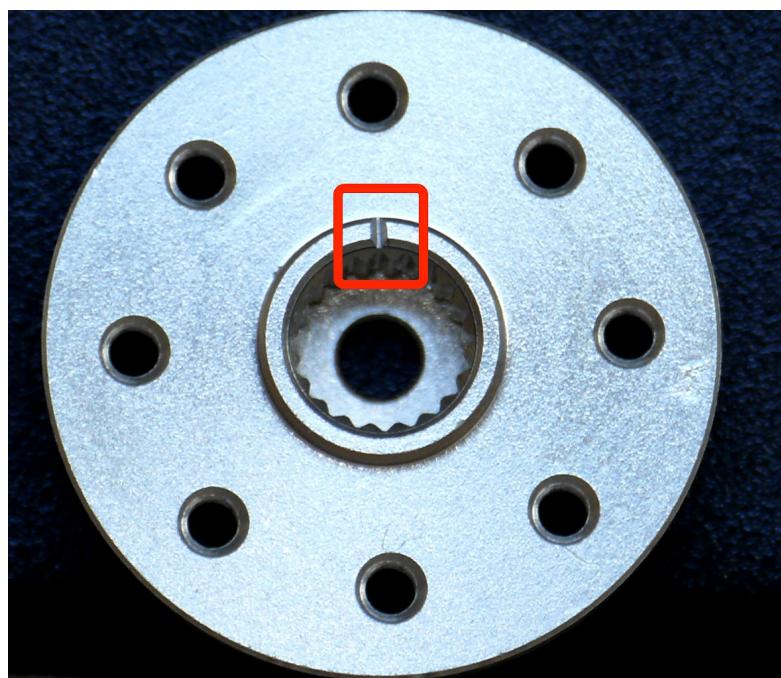
Construire un Torse Poppy ou un Humanoïde est plus complexe qu'un Ergo Jr mais ça n'est pas plus compliqué que de monter un Meccano ou un meuble Suédois. Cela consiste en les quelques étapes suivantes:

- assembler le support de chaque moteur: **soyez très attentif à la position zéro du moteur!**
- configurer les moteurs afin de se conformer à la *configuration poppy*
- utiliser de nombreuses vis pour connecter toutes les parties imprimées 3D aux moteurs
- faire un peu d'électronique pour la carte embarquée à l'intérieur de la tête : cela peut être un peu ardu si vous n'êtes pas familier avec l'électronique.

La **Patience et la précision** sont vos alliées, mais en cas d'erreur, ne paniquez pas: Poppy est un robot prévu pour être monté et démonté. Si vous êtes attentifs aux quelques **avertissemens** ci-dessous, et au prix de quelques essais et erreurs, vous aurez un Torse Poppy ou un Humanoïde en état de marche :

Avertissement 1: L'Humanoïde Poppy et le Torse Poppy sont construits avec des moteurs servos de type Dynamixel MX-28 et MX-64. Il s'agit de moteurs puissants et ils peuvent représenter un risque de dommage pour vos doigts et les matériaux autour. Soyez prudent et placez le robot dans un espace non encombré pendant les tests que vous effectuerez.

Avertissement 2: Alignez le point du support moteur ("horn") et le point sur l'axe du servo.



Avertissement 3: Ajustez les trois points des moteurs avec les trois points de la partie structurelle.



Avertissement 4: Utilisez du freinfillet afin d'éviter que les vibrations ne desserrent les vis. Par contre, tremper l'extrémité de la vis dans le freinfillet est suffisant (n'appliquez pas de freinfillet directement sur les trous de vis, cela serait trop et le démontage s'avèrerait très difficile!)

Guides d'assemblage pas-à-pas :

- [Guide pour l'Humanoïde](#)
- [Guide pour le Torse](#)

Démarrer et connecter le robot

Dans cette section, nous allons décrire comment démarrer votre robot et vous donner un aperçu des possibilités pour y accéder.

Configurer le logiciel

Les créatures Poppy viennent avec une carte embarquée dont le travail est de contrôler les moteurs et d'accéder aux capteurs. Pour une plus grande simplicité, cette carte embarquée peut être contactée à travers une interface Web. cela facilite le contrôle du robot depuis votre propre ordinateur ou une tablette sans avoir à télécharger ou installer quoique ce soit.

Il y a plusieurs méthodes pour configurer la carte embarquée de votre robot Poppy :

- **la méthode simple:** utilisez une image ISO toute prête et transférez la sur une carte SD
- **la méthode avancée:** tout installer vous-même

Si vous prévoyez d'utiliser un robot simulé, vous devez installer le logiciel sur votre propre ordinateur. Suivez les instructions pour [configurer la simulation](#).

Simplement: utiliser la Carte SD Poppy

Le moyen le plus simple et le plus rapide - de loin - est d'utiliser des images ISO déjà faites pour les cartes SD. Ces images ISO viennent avec tout le nécessaire déjà pré-installé pour votre robot Poppy. C'est aussi un bon moyen de vous assurer que vous utilisez exactement le même logiciel que nous Ainsi vous éviterez de nombreux problèmes.

Les kits de robotique Poppy viennent avec une carte SD prête à l'emploi. Ainsi vous n'avez rien de spécial à faire.

Les images ISO peuvent être trouvées avec les distributions de chaque créature :

- [pour le Poppy Ergo Jr](#)
- [pour le Torse Poppy](#)
- [pour l'Humanoïde Poppy](#)

Elles peuvent être écrites sur une carte SD d'au moins 8 Go en utilisant un outil classique. Une fois la carte SD prête, vous n'avez qu'à l'insérer dans votre carte embarquée. Vous pouvez ensuite mettre votre robot sous tension, il devrait démarrer et vous pourrez ainsi vous connecter à son interface web.

Plus de détails dans la [section Démarrage](#).

Plus avancé: tout installer vous-même

La méthode avancée consiste à installer tout le nécessaire depuis zéro. Elle suit la même procédure que nous utilisons pour générer les images ISO pour cartes SD. Nous mentionnons cette méthode car elle peut s'avérer utile si :

- **vous travaillez avec un robot simulé** et vous devez donc installer tout le logiciel nécessaire sur votre ordinateur, cette procédure est un bon endroit pour voir comment le faire sur un Raspberry Pi et l'adapter pour un autre ordinateur,
- vous voulez modifier l'environnement de travail,
- vous voulez simplement comprendre comment tout fonctionne.

Nous essayons de garder cette procédure d'installation aussi générique que possible. Pour autant, certains détails peuvent varier en fonction de votre système d'exploitation ou de votre ordinateur. De plus, l'installation depuis zéro demande quelques connaissances de base pour installer et configurer un environnement Python.

suivant ce que vous souhaitez accomplir, toutes les étapes ne sont pas nécessairement requises. En particulier, si vous souhaitez contrôler un robot simulé, il suffit d'installer les bibliothèques Python pour Poppy.

Plus de détails vous sont fournis dans le chapitre [Installation pour utilisateurs avancés](#).

Configurer le réseau

Une fois que votre Poppy est assemblé et son logiciel prêt, l'étape suivante est de le connecter à un réseau. Le but est de vous permettre d'accéder au robot depuis votre ordinateur ou votre tablette/smartphone, de le contrôler et de le programmer.

Il y a deux manières de connecter votre robot à votre ordinateur/tablette/smartphone:

- Connecter le robot et l'ordinateur au même réseau (par exemple la box wifi de votre domicile ou votre établissement scolaire).
- Connecter directement votre robot à votre ordinateur par l'intermédiaire d'un câble ethernet.

Bien que connecter le robot et l'ordinateur marche pour la plupart des utilisateurs, il semble que dans certains cas cela ne fonctionne pas.

Pour trouver l'adresse de votre robot sur le réseau, nous utilisons le [protocole standard Zeroconf](#). Il vous permet d'utiliser le nom d'hôte: "*poppy.local*" en tant qu'adresse. Cela devrait fonctionner sans aucune configuration sous Mac OS et GNU/Linux. Mais cela nécessitera d'installer les [Services d'impression Bonjour](#) sous Windows. Si vous préférez, vous pouvez utiliser l'adresse IP assignée à votre robot à la place. Si vous n'êtes pas administrateur de votre réseau cela peut être assez difficile à déterminer. Dans ce cas, utilisez de préférence la première procédure.

Pour vérifier que tout est configuré correctement, vous pouvez aller à l'adresse suivante avec votre navigateur web favori : <http://poppy.local/>. Vous pouvez remplacer *poppy.local* par l'adresse IP de votre robot (par exemple <http://192.168.0.42>).

Si vous n'êtes pas familier avec la configuration des réseaux ou si vous n'avez aucune idée de ce que le paragraphe précédent a tenté de vous expliquer, il est préférable de consulter un ingénieur réseau afin de compléter cette étape.

Utiliser l'interface Web

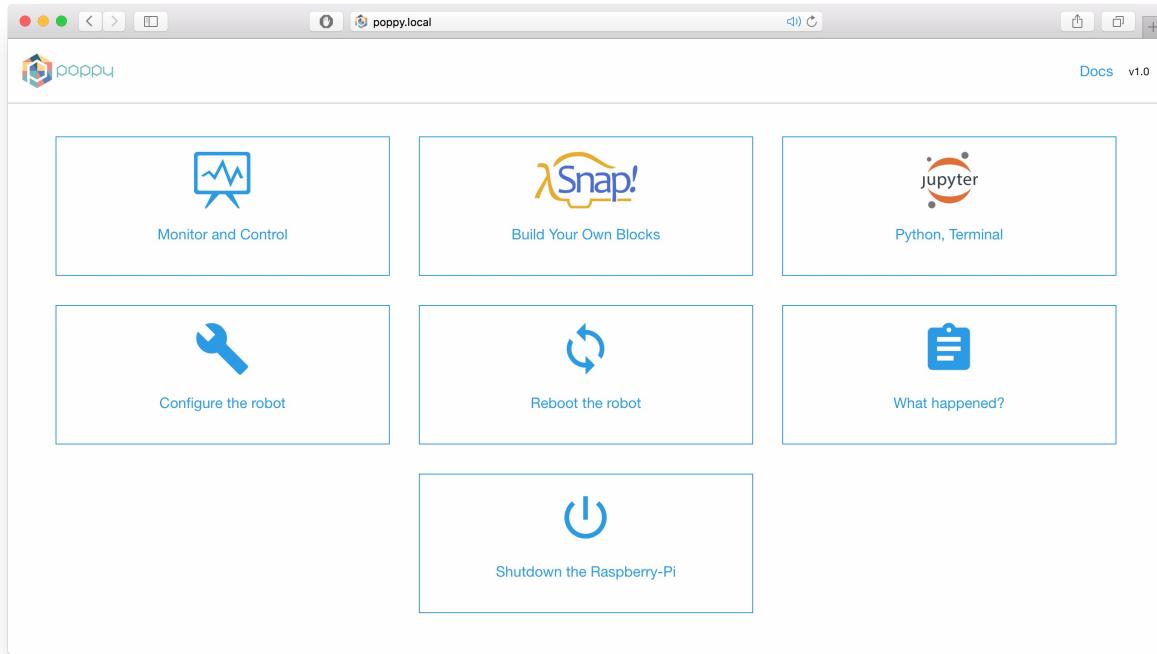
L'interface web est le point central pour contrôler, programmer et configurer votre robot. Elle peut être utilisée pour:

- Suivre l'activité du robot
- Le programmer en utilisant [Snap!](#)
- Le programmer en utilisant [Python](#)
- Configurer le robot (changer son nom, activer ou désactiver la caméra, le mettre à jour)
- Remettre à zéro et arrêter le robot.

Pour accéder à cette interface, il suffit d'aller à l'URL suivante avec votre navigateur Web favori:

- <http://poppy.local> (si vous avez changé le nom de votre robot, remplacez simplement *poppy* par son nouveau nom)
- ou utilisez son adresse IP.

Vous devriez voir quelque chose comme:

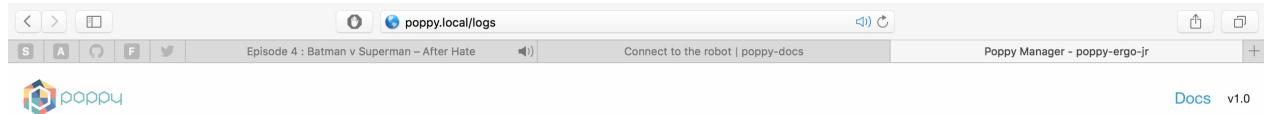


Les boutons peuvent être utilisés pour naviguer vers les différentes fonctionnalités. Par exemple, si vous cliquez sur le *Monitor and Control*, vous accédez à l'application de monitoring:



Cela vous permettra d'allumer ou d'éteindre les moteurs de votre robot, de suivre leur activité, et leur comportement d'allumage et d'extinction.

Le bouton *What happened* est là où vous devriez regarder pour plus d'information sur ce qui s'est mal passé. Voici une capture d'écran de ce que vous devriez voir si tout se passe comme prévu:



Logs

```
Attempt 1 to start the robot...
SnapRobotServer is now running on: http://0.0.0.0:6969

You can open Snap! interface with loaded blocks at "http://snap.berkeley.edu/snapsource/snap.html#open:http://10.204.4.89

HTTPRobotServer is now running on: http://0.0.0.0:8080

Robot created and running!
```


Programmer le robot

Les robots Poppy sont conçus pour être faciles à programmer. Trois options sont présentées ici:

- avec [Snap!](#), une variante de Scratch, un langage de programmation visuel,
- avec [Python](#) pour bénéficier de toute la puissance de l'interface de programmation,
- à travers l'[interface REST](#) qui vous permet de contrôler les robots Poppy avec d'autres équipements et n'importe quel langage de programmation.

Comme pour le reste du projet, toutes nos bibliothèques sont sous licence libre et disponibles sur [GitHub](#).



Utiliser Snap



Snap! est un langage de programmation visuel - une variante du célèbre langage Scratch. C'est un langage de programmation basé sur des blocs à connecter ensemble, permettant une introduction très complète à l'informatique. Il fonctionne dans votre navigateur Web et est implémenté en Javascript. Nul besoin d'installer quoique ce soit afin de l'utiliser. Il existe sous licence libre et est activement maintenu.

Nous avons développé un ensemble de blocs spécifiques pour les robots Poppy qui permettent d'envoyer des commandes moteur et de lire des valeurs en provenance des capteurs de votre robot. Cela vous permet de vous immerger dans le contrôle et la programmation de votre robot sans problèmes de compilation ou de syntaxe. Grâce à la boucle d'interaction continue de Snap! il suffit de cliquer sur un bloc pour envoyer la commande associée au robot. Snap! s'applique aussi tout naturellement aux projets plus complexes.

Un [chapitre dédié](#) vous guidera dans ce que vous pouvez faire avec Snap! et les robots Poppy.

Utiliser Python



Les bibliothèques Poppy ont été écrites en Python, pour permettre un développement rapide et extensible, et pour bénéficier de toutes les bibliothèques scientifiques existantes. Python est aussi un langage très répandu et largement utilisé dans les sphères pédagogiques et artistiques. En programmant Poppy en Python, vous aurez accès tant aux fonctions de bas niveaux qu'à celles de plus haut niveau.

L'interface de programmation (API) a été conçue pour le prototypage rapide. Créer un robot et commencer à mouvoir ses moteurs ne devrait pas prendre plus de quelques lignes:

```
from poppy.creatures import PoppyErgoJr  
  
jr = PoppyErgoJr()  
jr.m3.goal_position = 30
```

Nous sommes aussi de grands fans du [projet Jupyter](#) et de ses "Notebooks". Les "Notebooks" sont des documents qui peuvent aussi bien contenir du code Python et du texte formaté que des équations, des images ou des vidéos. Ils peuvent être édités depuis l'interface Web Jupyter ce qui permet aux utilisateurs de programmer les robots Poppy depuis un site web hébergé sur l'ordinateur même du robot. Nous pensons qu'il s'agit là d'un outil puissant pour permettre la création et le partage de code vivant, de visualisations des résultats et du texte explicatif combinés en un seul document.



La plupart des tutoriels, des expériences et des activités pédagogiques que nous et notre communauté développons sont disponibles sous forme de Notebooks Jupyter.

The screenshot shows a Jupyter Notebook interface with the title "Untitled" and "Last Checkpoint: 9 minutes ago (unsaved changes)". The toolbar includes File, Edit, View, Insert, Cell, Kernel, Help, and Cell Toolbar: None. A Python 2 logo is in the top right. The main content area has a heading "Discover the Poppy Ergo Jr" and a photograph of the Poppy Ergo Jr robot, which is a white lamp-like creature with articulated arms and legs. Below the image, text says: "In this notebook, you will learn the first steps for controlling an Ergo Jr. In particular, you will see how:" followed by a bulleted list: • Create in Python the robot • Learn how to control its motors • Read values from the motors (such as position). A section titled "Create the Robot in Python" follows, with text: "First, you have to import the class representing your creature from the poppy libraries:" and a code cell: "In []: `from poppy.creatures import PoppyErgoJr`". Below it, text says: "Then, you can directly "instantiate" it:" and another code cell: "In []: `ergo_jr = PoppyErgoJr()`".

Une galerie de Notebooks mise à jour peut être trouvée [ici](#). Vos contributions sont les bienvenues!

Avec l'interface REST

En plus des options Snap! et Python, nous voulions fournir un autre moyen d'accéder et de contrôler votre robot depuis n'importe quel appareil ou langage de programmation. Les robots Poppy exhibent une interface REST. Les fonctionnalités principales du robot sont donc accessibles par des requêtes HTTP GET/POST.

D'un point de vue plus pratique, cela vous permet de pouvoir:

- **Ecrire des connecteurs pour contrôler les robots Poppy depuis n'importe quel langage de programmation** (de nombreux contributeurs ont déjà écrit des connecteurs [Matlab](#) et [Ruby](#)).
- **Concevoir des applications web** connectées à votre robot, telles que cette [interface de suivi](#) (aussi une contribution!).
- Faire interagir votre **robot avec d'autres appareils connectés** tels qu'un smartphone, des capteurs intelligents, ou même votre compte Twitter...

L'interface REST est encore en cours de réalisation, elle va certainement changer et est actuellement en manque de documentation ! Pour plus d'informations, veuillez consulter [cette page](#) ou notre [forum](#). Une interface REST bien conçue, stable et bien documentée est attendue pour la prochaine version majeure de notre logiciel.

Visualiser le robot dans un simulateur

Créatures Poppy simulées

Des versions simulées de tous les robots Poppy (Humanoïde, Torse, and Ergo Jr) sont disponibles.

Des connexions avec deux simulateurs populaires ont été développées :

- avec [V-REP](#): une plateforme d'expérimentation robotique virtuelle
- avec [un visualisateur web 3D](#): plus léger mais sans support pour la physique.

Actuellement, seul le Poppy Ergo Jr peut être utilisé dans le visualisateur web. Si vous souhaitez simuler d'autres créatures, vous devez utiliser V-REP. Le support pour d'autres robots est en préparation mais pas dans un avenir immédiat.

Nous pensons que la simulation peut être un outil puissant. Elle permet de développer et de tester des programmes sans avoir besoin d'un vrai robot. Cela est particulièrement utile:

- Pour découvrir et essayer les possibilités du robot sans dépenser quoique ce soit.
- Dans un contexte où plusieurs utilisateurs doivent partager un robot. Par exemple dans une salle de classe où chaque groupe peut travailler avec le simulateur puis valider son travail sur un vrai robot.
- Pour concevoir et exécuter des expériences compliquées et coûteuses en temps.

Nous essayons de rendre la **transition du robot Poppy simulé vers le vrai robot aussi transparente et simple que possible**. Notre documentation est tout à fait valide à la fois pour le robot simulé et le vrai robot. Le chapitre [De la simulation au vrai robot](#) vous guidera dans les quelques étapes nécessaires pour transformer votre programme de simulation vers un programme fonctionnant sur un vrai robot.

Si vous voulez utiliser les robots Poppy dans un simulateur, il faut installer quelques bibliothèques Poppy sur votre ordinateur.

Installer le logiciel requis

Bien que les robots physiques viennent les bibliothèques pré-installées, elles ne sont pas intégrées dans les simulateurs. Vous devrez donc les installer sur votre ordinateur. Plus de détails sur ce que vous aurez à faire vous est donné dans la section ci-dessous. Vous n'aurez pas non plus accès à l'interface web du robot. Vous devrez lancer manuellement les différents services pour commencer à programmer votre robot (par exemple le serveur Jupyter pour les Notebooks en Python, ou le serveur Snap!).

Nous espérons fournir une application facile à installer pour Windows/Mac/Linux fournissant tout le nécessaire à une date ultérieure, mais pas dans un futur immédiat.

Pour commencer à contrôler votre robot Poppy de simulation, avec soit V-REP ou le visualisateur Web, vous aurez besoin de:

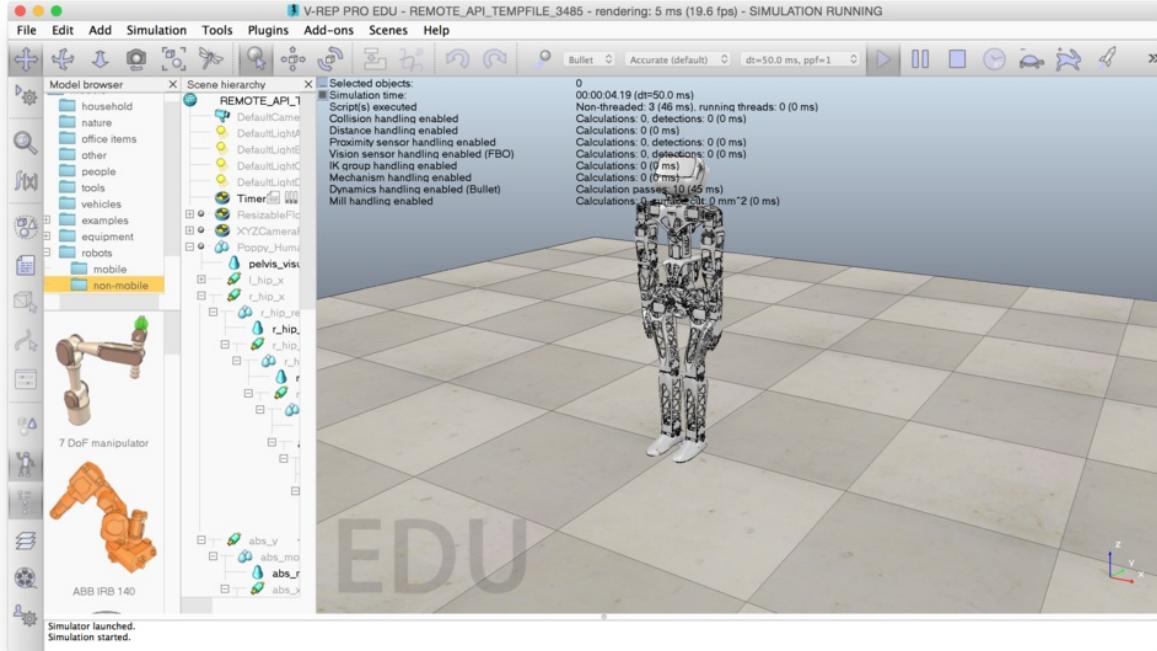
- Obtenir un environnement Python fonctionnel, nous vous encourageons à utiliser la [distribution Anaconda Python](#). Elle fonctionne avec toute version ≥ 2.7 ou ≥ 3.4 . Préférez la version Python 2.7 si vous pouvez car il s'agit de la version que nous avons utilisée.
- Installer les bibliothèques Poppy : pybot et la bibliothèque correspondant à votre créature (par exemple poppy-ergo-jr).

Des détails sur ces étapes peuvent être trouvées dans la section [Installer localement le logiciel pour usage avec un simulateur](#).

Utiliser V-REP

[V-REP](#) est un simulateur puissant et très répandu. Il est largement utilisé à des fins de recherche et d'enseignement. Il est de plus disponible gratuitement sous une licence éducative. Il peut être téléchargé depuis [ce site web](#) (il fonctionne sous Mac OS, Windows et GNU/Linux).

Il est important de noter que V-REP simule tous les aspects de la physique et de l'affichage du robot, il peut être lent si vous ne disposez pas d'un ordinateur puissant (notamment en terme de carte graphique).

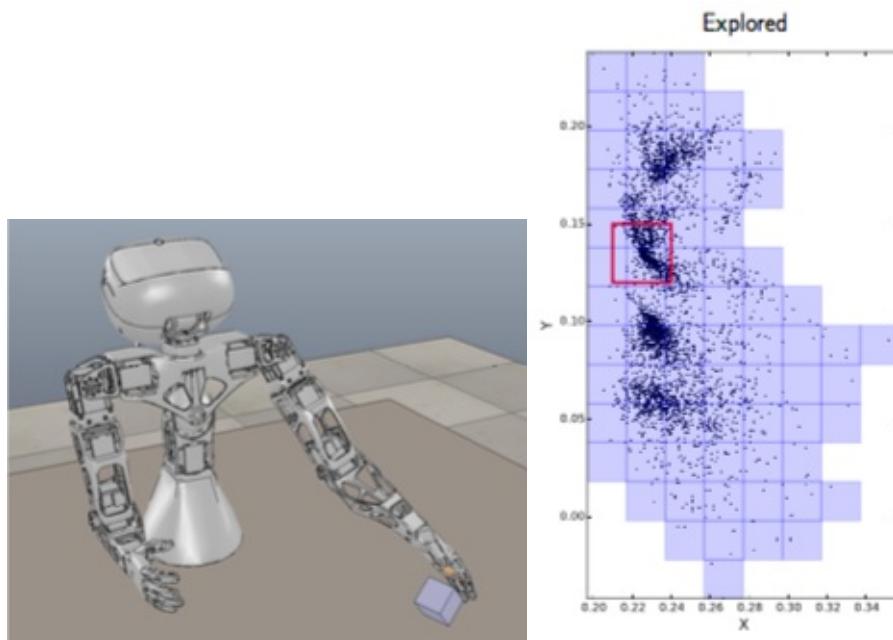


Les principaux robots Poppy sont disponibles dans V-REP:

- Humanoïde Poppy
- Torse Poppy
- Poppy Ergo Jr

V-REP peut être utilisé pour apprendre à contrôler les moteurs, obtenir de l'information des capteurs mais aussi interagir avec un environnement simulé. Il peut être contrôlé avec Python, Snap! ou par l'interface REST. Voici quelques exemples de ce que la communauté a pu faire avec:

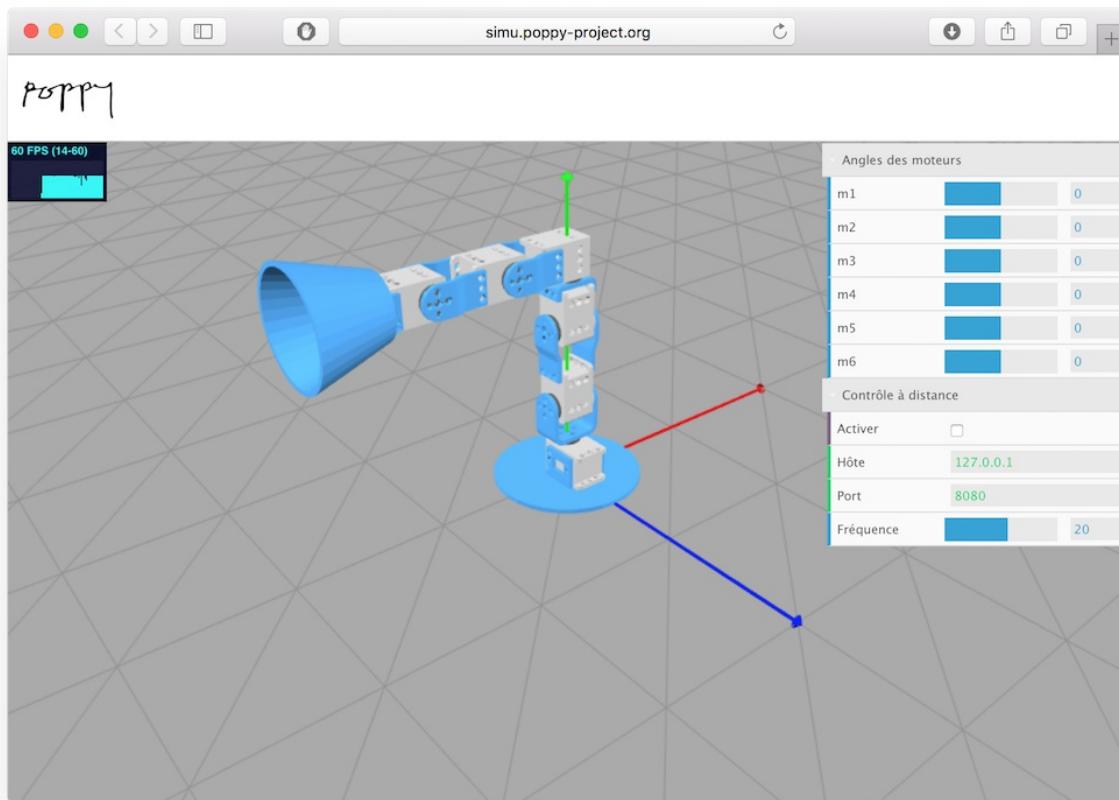
- Une activité pédagogique pour découvrir les différents moteurs de votre robot et comment les contrôler.
- Une expérience scientifique, dans laquelle un Torse Poppy apprend à pousser le cube qui se trouve sur la table en face de lui.



Malgré nos efforts pour reproduire le comportement et le fonctionnement du robot, quelques différences subsistent. En particulier, si vous faites marcher le robot en simulation, cela ne signifie pas forcément qu'il fonctionnera dans le monde réel (et vice-versa).

Utiliser notre visualisateur web

Notre visualisateur web - basé sur la bibliothèque [Three.js](#) - vous montrera une représentation 3D de votre robot Poppy. Pour cela vous aurez besoin de le connecter soit à un vrai robot (à travers l'interface REST) ou à un robot de test tournant sur votre ordinateur. Il suffit de régler la variable hôte avec l'adresse de votre robot depuis l'interface Web.



Un robot de test peut être démarré avec la commande **poppy-services**. Par exemple: `poppy-services --poppy-simu --snap poppy-ergo-jr`

Comme pour V-REP, vous pouvez contrôler votre robot en utilisant Python, Snap!, ou l'interface REST. En revanche, il n'y a pas de simulation physique, cela demande donc moins de ressources mais vous ne pourrez pas interagir avec des objets.

Voici un exemple avec Python:

The screenshot shows two side-by-side browser windows. The left window is titled 'Visualisateur Poppy Ergo Jr' and displays the text 'Poppy Ergo Jr pour le navigateur'. Below this, there is usage information and two blue buttons: 'OUVRIR LE VISUALISATEUR' and 'OUVRIR L'ÉDITEUR SNAP!'. The right window is titled 'jupyter' and shows a file tree with 'visu-demo' selected. It displays the message 'Notebook list empty.' at the bottom.

simu.poppy-project.org Visualisateur Poppy Ergo Jr

Poppy Ergo Jr pour le navigateur

Pour utiliser le visualisateur, vous devez avoir une instance d'une créature en cours d'exécution.
Si vous ne connaissez pas la marche à suivre, veuillez vous référer à [la documentation](#).

You devez aussi utiliser un [navigateur supportant le WebGL](#), comme Firefox ou Chromium (Chrome).

OUVRIR LE VISUALISATEUR

OUVRIR L'ÉDITEUR SNAP!

jupyter

localhost Desktop/visu-demo/

Select items to perform actions on them.

Files Running Clusters

/ Desktop / visu-demo ..

Notebook list empty.

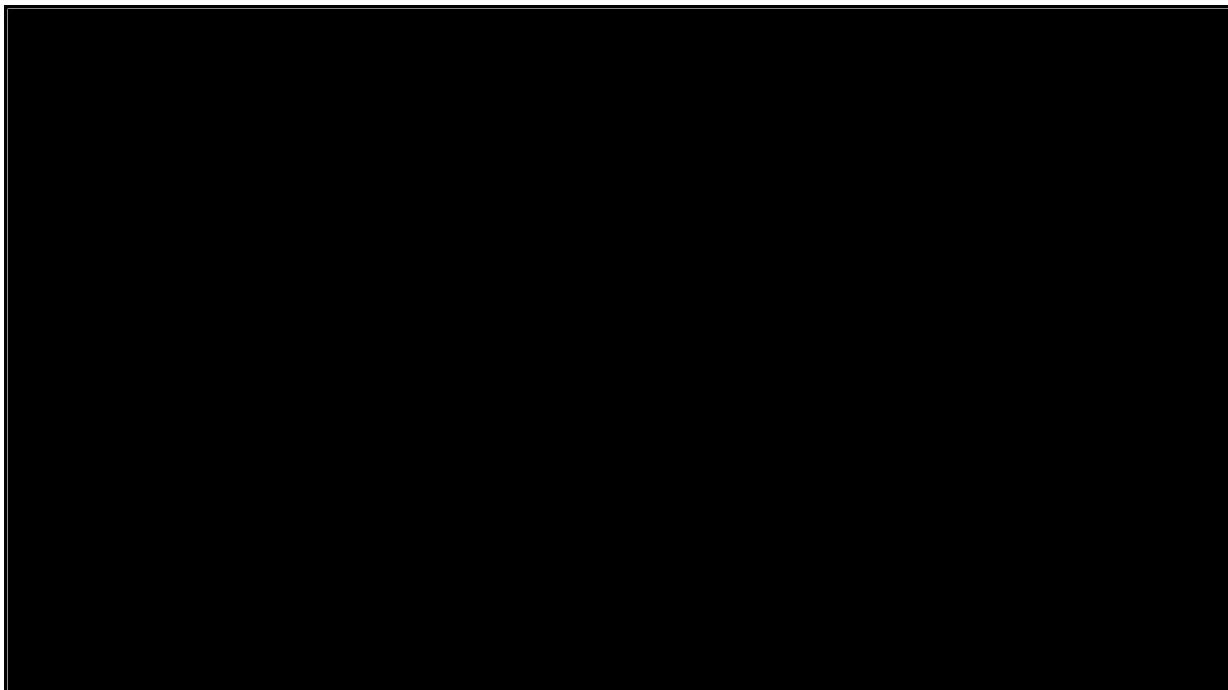
Aperçu des projets développés par la communauté

La communauté Poppy rassemble une communauté interdisciplinaire de débutants et d'experts, de scientifiques, d'éducateurs, de développeurs et d'artistes. De nombreuses créations robotiques ont émergées. En voici quelques-unes illustrées ci-dessous.

School of Moon

La pièce [School of moon](#) a été créée par la troupe de danse contemporaine [Shonen](#) conduite par le choréographe Eric Minh Cuong Castaing.

La scène est partagée avec des enfants, deux danseurs (Gaëtan Brun Picard and Ana Pi), 3 Robots Nao et deux robots Humanoïdes Poppy. Cette pièce est une métaphore de la création d'une post-humanité en trois actes: l'Homme, l'Homme et la Machine, et la Machine.



Les représentations sont adaptées à l'endroit où elles se tiennent, les enfants participants à la danse appartiennent à la communauté locale. Il y a aussi des séquences spécifiques suivant les robots disponibles dans la ville.

Les défis artistiques sont:

- de diriger des enfants sur scène
- d'avoir des interactions entre humains et robots sur scène
- d'avoir des robots sur scène.

La création s'est tenue sur quatre périodes de temps:

- 2 semaines en résidence au CDC de Toulouse (France) en Septembre 2015
- 2 semaines en résidence au Klap de Marseille (France) en Décembre 2015
- 4 semaines en résidence au *Ballet National de Marseille* (France) en Janvier 2016
- 2 semaines en résidence à Düsseldorf (Germany)

Le Projet Cherry

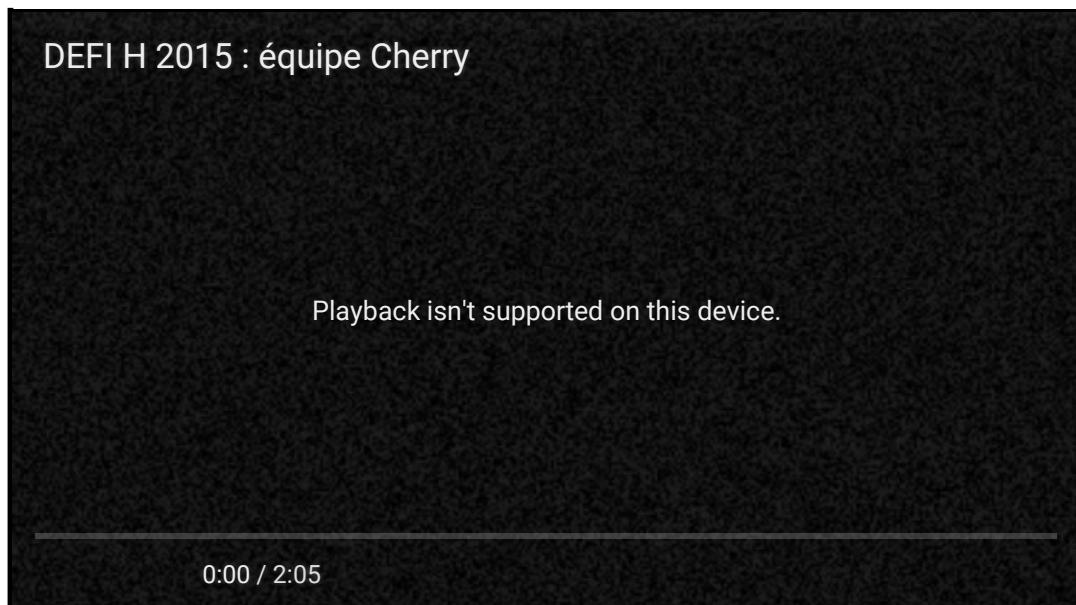


Le projet Cherry est un projet communautaire pour développer des scénarios afin de réduire l'isolation des enfants à l'hôpital.

Ce projet utilise le robot Poppy comme compagnon pour les enfants hospitalisés. Cherry peut compenser la rupture sociale durant l'hospitalisation. Il agit comme médiateur entre l'enfant, ses amis, sa famille et les enseignants, et peut lui parler ou jouer avec lui.

Il agit aussi au niveau pédagogique, en encourageant l'enfant à interagir avec l'école, en offrant des quizz et des jeux éducatifs.

Enfin, un dernier aspect est d'assister le personnel médical dans l'éducation thérapeutique. En effet, un message est parfois plus acceptable pour l'enfant s'il est délivré par un robot, plutôt que par un adulte habillé comme un médecin.



Plus d'information:

- La page [facebook](#)
- Le flux [twitter](#)
- Le blog [wordpress](#)
- Le dépôt [github](#) (avec un [wiki](#))

Connecter Poppy et Arduino grâce à Snap4Arduino

Gilles, enseignant et "maker" à ses heures perdues, a développé de nombreux projets basés sur le Poppy Ergo Jr et Arduino. Pour connecter les deux mondes, il utilise [Snap4arduino](#). Alors, il devient très aisément élégant de les faire communiquer. Vous pouvez combiner Arduino avec des blocs Poppy et voilà !! Vous pouvez contrôler votre robot avec n'importe quel capteur basé sur Arduino.

PoppyErgoJr + Snap4Arduino + HC-SR04

Playback isn't supported on this device.

0:00 / 0:19

La seule limite est alors votre créativité! Par exemple, vous pouvez faire [Poppy Ergo Jr jouer au Morpion](#):

PoppyErgoJr playing Tic-Tac-Toe, Snap4Arduino ...

Playback isn't supported on this device.

0:00 / 1:25

La documentation détaillée peut être trouvée dans la section [Contrôler Poppy avec Arduino](#).

Assembly guides

This section will provide step by step documentation of various libraries used in Poppy robots.

- [Assemble the Ergo Jr](#)
 - [Electronic assembly](#)
 - [Motor configuration](#)
 - [Hardware construction](#)
- [Assemble the Poppy Humanoid](#)
- [Assemble the Poppy Torso](#)

Assembly guide for the Ergo Jr



The Poppy Ergo Jr robot is a small and low cost 6-degree-of-freedom robot arm. It consists of very simple shapes which can be easily 3D printed. They are assembled via OLLO rivets which can be removed and added very quickly with the OLLO tool.

Its end effector can be easily changed. You can choose among several tools:

- a lampshade,
- a gripper,
- or a pen holder.

Thanks to the rivets, they can be very quickly and easily swapped. This allows the adaptation of the tooltip to the different applications you plan for your robot.

The engines have the same functionality as other Poppy creatures but are slightly less powerful and less precise. The advantage being that they are also less expensive.

The electronic card is visible next to the robot, which is very advantageous to understand, manipulate, and plug extra sensors. No soldering is needed, you just need to add the shield for XL-320 motors on top of the Raspberry Pi pins.

This chapter will guide you through all steps required to entirely assemble a Poppy Ergo Jr. It will cover:

- [motors configuration](#)
- [electronic assembly](#)
- [hardware construction](#)

The entire assembly should take about one or two hours for the first time you build one. With more practice, half an hour should be more than enough.

At the end of the process, you should have a working Poppy Ergo Jr, ready to move!

We recommend you to follow carefully the instructions. Even if the Ergo Jr can be easily disassembled, it is always disappointing to need to start from scratch again because you forgot to configure the motors, a wire is missing, or a motor is reversed.

Bill of materials

Here you will find the complete list of material (BOM) needed to build a Poppy Ergo Jr.

Poppy's material

- 1x [Pixel board \(electronic board to control XL320 motors\)](#)
- 1x [disk_support.stl](#) (using laser cutting) the 2D plan can be found [here](#). You can also 3D print the base but it will take a lot of time
- the 3D printed parts [STL here](#)
 - 1x [base.stl](#)
 - 3x [horn2horn.stl](#)
 - 3x [side2side.stl](#)
 - 1x [long_U.stl](#)
 - 1x [short_U.stl](#)
 - 1x [support_camera.stl](#)
 - the different tools
 - 1x [lamp.stl](#)
 - 1x [gripper-fixation.stl](#)
 - 1x [gripper-fixed_part.stl](#)

- 1x gripper-rotative_part.stl
- 1x pen-holder.stl
- 1x pen-screw.stl

Robotis parts

- 6x Robotis dynamixel motors XL-320
- 1x set of OLLO rivets (about 70 colored and 4 grey)
- 1x OLLO TOOL

Screw

- 4x M2.5x6mm screw (for fixing the Raspberry Pi on the base)
- 4x M2x5mm screw (for fixing the camera)
- 4x M2 nuts (fixing camera)
- 1x Standoff Male/Female M2.5 10mm

Various electronics

- 1x Raspberry Pi 2
- 1x micro SD 8Go
- 1x Raspberry Pi camera
- 1x AC power 7.5V 2A with a 2.1 x 5.5 x 9.5 jack connector ([this one](#) for instance).
- Short ethernet cable

Electronic assembly

Insert the micro-SD card in the Raspberry Pi

Make sure you use a pre-configured micro SD-card. If it not the case, you have to "burn" your micro-SD card with the ergo-jr ISO image, this is described in the [startup section](#).

Insert the micro-SD card inside the Raspberry Pi: push the micro-SD in the connector slot until you hear a "click" sound.



Assemble the pixl board

The pixl board will be available for purchase in May 2016 at Generation robot.

[Pixel bord](#) allow to power up the Raspberry Pi from 7.5V DC power or battery and communicate with XL-320 motors.

Plug the pixl at the end of Raspberry Pi headers.

Once the pixl is plugged (**and not before**), you can plug the DC power and the motors wire.

You need to switch off the power supply of the Pixl board before to put in or to take off the Pixl board of the Raspberry pi.

Otherwise, you risk to burn the power converter of the Pixl board.

You can now [configure your motors](#).

Motor configuration

The Ergo Jr is made of 6 XL-320 motors from [Robotis](#). Each of this servomotor embeds an electronic board allowing it to receive different kind of orders (about position, speed, torque...) and to communicate with other servos. Therefore, you can chain up several of this servomotors and command them all from one end of the chain: each servomotor will pass the orders to the next one.



Yet, in order for the motors to be connected and identified on the same bus (same line), they must have a unique ID. Out of the factory they all set to the same ID: 1. In this section, we will give you details on how you can set a new and unique ID to each of your motors.

We recommend to configure motors in parallel of the hardware assembly. Meaning, that before assembling a new motor, you first configure it, then assemble to the rest of your robot. This will prevent you to swap motors. In the step-by-step assembly procedure, we will point out each time you need to configure a new motor. Furthermore, you will also be able to directly configure the motor from the assembly notebook interface.

Configuring motors one at a time

As explained above, all motors have the same ID by default. **Only one motor at a time should be connected to the data bus when you configure them.** Otherwise, it will not work as all motors connected will think that the order sent on the line is intended for them, they will all try to answer resulting in a big mess.

Your electronic setup when configuring a motor should look like this:

- the Raspberry Pi
- the shield on top and the AC plugged
- a wire from the shield to the motor you want to configure
- a ethernet cable going from the Raspberry Pi to your computer or your router

Web interface (user friendly)

The web utility for configuring motors is still in the TODO stack.

Command-line utility

Robots come with a command line utility `poppy-configure` ; to use it you need to open a terminal on your Raspberry Pi.

You can access to the Raspberry Pi directly from your computer. To do so, open the page <http://poppy.local> in a web browser. You see the Poppy home page. Click on the "Python, Terminal" link and select "New Terminal".

Once the terminal is open, copy and press enter to execute the command below.

```
poppy-configure ergo-jr m1
```

You have now configured the m1 motor of your robot. Once configured and that you see the message indicating that everything went well, you can unplug the motor (you don't need to turn off the card). The configuration of the motor is stored in the motor internal memory.

Poppy Ergo Jr motors are named m1, m2, m3, m4, m5, m6. To configure the others motors, you have to change m1 by the name of the motor you want to configure in the command above.

Mechanical Assembly

General advices and warnings

- You can assemble all the rivets before the construction. You have to put the edges of the first part in the second part holes. You will thus be able to remove them easily if needed.



- There are two kinds of rivets. The grey ones and the others. Grey rivets are longer to be able to be inserted in the motor axis, at the opposite side of the horn.



- Use the OLLO Tool for putting and removing rivets easily.



- Do not forget to put wires between motors while building the robot! Each motor, except the last, must have two wires; one connected to the previous motor and the other to the next (no favourite side).

- **Always align the horn with the motor before assembling them!** Otherwise your Poppy Ergo Jr will look all weird.



- Every motor horns (black revolving circle) are **facing the left side of the robot**. It is a convention but it will define the orientation of your motors.



Step-By-Step guide

Motor configuration (for all steps)

You can configure your motors before, during or after the the mechanical assembly but it is highly advised to configure each motor one by one in the construction order :

- configure motor m1
- assemble the base and motor m1
- configure motor m2
- ...

To configure motors, you have to connect them separately one by one to the Raspberry Pi. If you try to configure a new motor wired to a previously configured motor, this will not work.

Please consult the [motor configuration section](#) for more informations.

Step 1

First, [configure one XL-320](#) motor as "m1".

Mount the motor on the 3D printed base.



To do so, prepare 8 small rivets. put the first part in the second part without putting them in the motor. Then, place the motor in the base, with the horn facing up and near the more open side. Use the Ollo to grab a rivet between the first and the second part, then put the rivet in one the assembly holes. Once the rivet is in place, lock it by pushing the part 1 of the rivet in part 2.

Step 2

Configure the second motor, its name is "m2", with the following command in a poppy terminal:

```
poppy-configure ergo-jr m2
```

Mount the *long_U* part. Be careful wih the orientation of the U, the horn must be oriented in the left. Mount the motor "m2" on top of the construction.



Step 3

Configure a third motor: "m3".

Mount *horn2horn* and *horn2side* parts on motor "m2", and mount "m3" on top of the construction.



Step 4

Configure the fourth motor: "m4".

Mount the *short_U* on it.



Mount motor "m4" and the assembled *short_U* on top of the previous assembly. The nose of the motor should be on the other side of the base.



Step 5

Configure the fifth motor: "m5".

Mount *horn2horn* and *horn2side* parts on motor "m4", and mount "m5" on top of the construction.



Step 6 - the tool of your choice

Configure the sixth motor: "m6".

To finish your Ergo Jr, you need to add a tool at its end. So first choose the tool you want depending on what you want to do.

Tools they can be easily and quickly changed, so you can adapt it to the different activities.

Lampshade or pen holder

Mount *horn2horn* and *horn2side* parts on motor "m5", and mount "m6" on top of the construction.



You can mount the pen holder or the lampshade on the motor "m6".



Gripper

Mount the *gripper-fixation* between motors "m5" and "m6".

Mount *gripper-fixed_part* and *gripper-rotative_part* on motor "m6".

Step 7 - electronics

Mount the support_camera part on the base. Fix the Raspberry Pi camera on it and move the camera flex cable between motor "m1" and the base.

To fix the flex cable of the camera on the Raspberry Pi:

- open the camera connector by pulling on the tab
- make sure that connectors on the flex cable are facing away of the ethernet port
- push the flex on the port, and push the plastic tab down

Motors wires:

If it is not already done, you can plug every motors wires. Every motor has two connectors but there is no input or output: you just have to create a chain of motors. The first motor is linked to the pixel and the second motor; the last motor is linked only to the previous one, and every other motors are linked to the one above and ahead.

Connectors of the motor "m1" (in the base) are a bit hard to link, you can use the OLLO tool to help yourself.



Step 8 - fix you ergo-jr to wood disk support

Mount you ergo-jr to the wood *disk-support*.

Mount the Raspberry Pi to the disk support, and use 4 x M2.5x6mm screw to fix it.

Done



Grab your [favorite drink](#) and relax.

