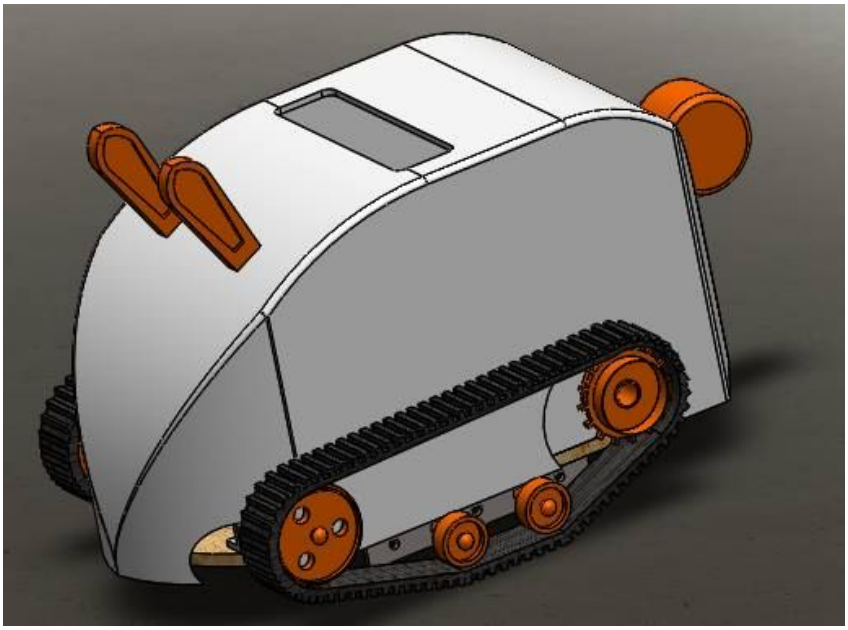
INSA INSTITUT NATIONAL DES SCIENCES APPLIQUÉES ROUEN

Projet de Physique P6
STPI/P6/2017 - 26

Robot mobile de surveillance piloté par WIFI à partir d'un smartphone



Etudiants :
Antoine BAILLIOT Guillaume DRON
Cécilia MALONGA Corentin PORIEL
Damien TOOMEY Juliette VALLOT

Enseignant-responsable du projet :
Monsieur Fabrice DELAMARE

Date de remise du rapport : 19/06/2017

Référence du projet : STPI/P6/2017 – 26

Intitulé du projet : Robot mobile de surveillance piloté par WIFI à partir d'un smartphone

Type de projet : Expérimental

Objectifs du projet :
Réalisation d'un robot piloté par WIFI.
Conception du robot sur SolidWorks et impression 3D.
Programmation d'une application Android avec MIT App Inventor.
Modification du programme Arduino fourni par le professeur.
Soudure de composants électroniques sur un circuit électronique.

Mots-clefs du projet : SolidWorks, Impression 3D, Application Android, Cartes electroniques

Table des matières

1.Introduction……...……5

2.Méthodologie / Organisation du travail……...….6

3.Travail réalisé et résultats…….…7

3.1. Partie mécanique……...……..…7

3.1.1. Conception de la forme générale du robot……....…7

3.1.2. Modélisation sur SolidWorks…….....…8

3.1.3. Fabrication des pièces et montage du robot…….…10

3.1.4. Dimension esthétique……...…12

3.2. Application Android et carte NodeMCU V3……...……13

3.2.1. Code MIT App Inventor……...…15

3.2.1.1. Ecran 1……...……16

3.2.1.2. Ecran 2……...……19

3.2.1.3. Ecran 3……...…22

3.2.2. Code Arduino……...…25

3.2.2.1 Code Arduino - méthode de développement……..……...25

3.2.2.2 Problème rencontré……...….25

3.3. Partie électronique……...……26

3.3.1 Travail réalisé…….…...…….....……....…..…26

3.3.2 Problèmes rencontrés…….……...……28

3.3.3 Résultat……....…….…29

4. Conclusions et perspectives……...…30

5. Bibliographie……...……32

6. Annexe ……...……...……34

NOTATIONS, ACRONYMES

URL : Uniform Resource Locator
SSID : Service Set IDentifier
HA : Horizontal Arrangement
TB : TextBox
IDE : Integrated Development Environment
CAO : Conception Assistée par Ordinateur

1.Introduction

Nous sommes actuellement en STPI 2 et nous n'avons pas tous choisi les mêmes thématiques au quatrième semestre. Certains ont pris ASI/GM, tandis que d'autres ont opté pour MECA/ASI, MECA/MRIE ou encore EP/MECA.

Le but du projet était de réaliser un robot de surveillance piloté par WIFI. Nous pouvons donc distinguer plusieurs objectifs. Premièrement, nous devions faire la conception du capot du robot sur SolidWorks puis l'imprimer en 3D et utiliser des pièces de récupération pour le reste du robot. Il était également nécessaire de souder les composants électroniques adaptés sur une carte imprimée. Enfin, nous devions réaliser une application Android qui permettrait de communiquer d'une part avec la carte NodeMCU V3 pour piloter le robot et d'autre part avec un smartphone placé sur le robot pour récupérer le flux vidéo grâce à la caméra de celui-ci. Le tout devait s'effectuer en WIFI.

Même si au départ nous n'avions pas les connaissances nécessaires pour effectuer un tel projet, nous nous nous sommes formés grâce à des tutoriels présents sur internet ainsi qu'avec les conseils avisés de Monsieur DELAMARE.

2.Méthodologie / Organisation du travail

Concernant l'organisation, nous nous sommes séparés les tâches en trois groupes de deux. Cécilia et Juliette ont décidé de s'occuper de la partie modélisation et impression 3D car elles avaient déjà travaillé sur SolidWorks. Au départ, Guillaume et Corentin devaient s'occuper de la partie soudure des composants électroniques, Antoine et Damien de la partie programmation de la NodeMCU V3 et de l'application Android mais Damien s'est confronté à de nombreux problèmes pour commencer à programmer sur Android Studio (un logiciel permettant de coder des applications Android) notamment pour installer le logiciel puis pour coder en Java, langage abordé en ASI et en GM. Damien a donc décidé de réaliser l'application sur MIT App Inventor, un logiciel en ligne qui permet de créer des applications Android de manière plus intuitive, sans taper des lignes de codes. Monsieur Delamare nous a donné le code pour la NodeMCU V3 afin gagner du temps.

Finalement, Antoine et Damien ont travaillé sur l'application et Guillaume et Corentin se sont intéressés au code de la NodeMCU V3 afin de le comprendre et de le modifier pour l'adapter à notre projet. Nous avons donc décidé de mettre la partie électronique de côté pour quelques séances.

Puis lorsque la carte imprimée nous est parvenue, Corentin et Damien ont continué la programmation tandis que Antoine et Guillaume se sont occupés de la soudure des composants sur la carte imprimée.

3.Travail réalisé et résultats

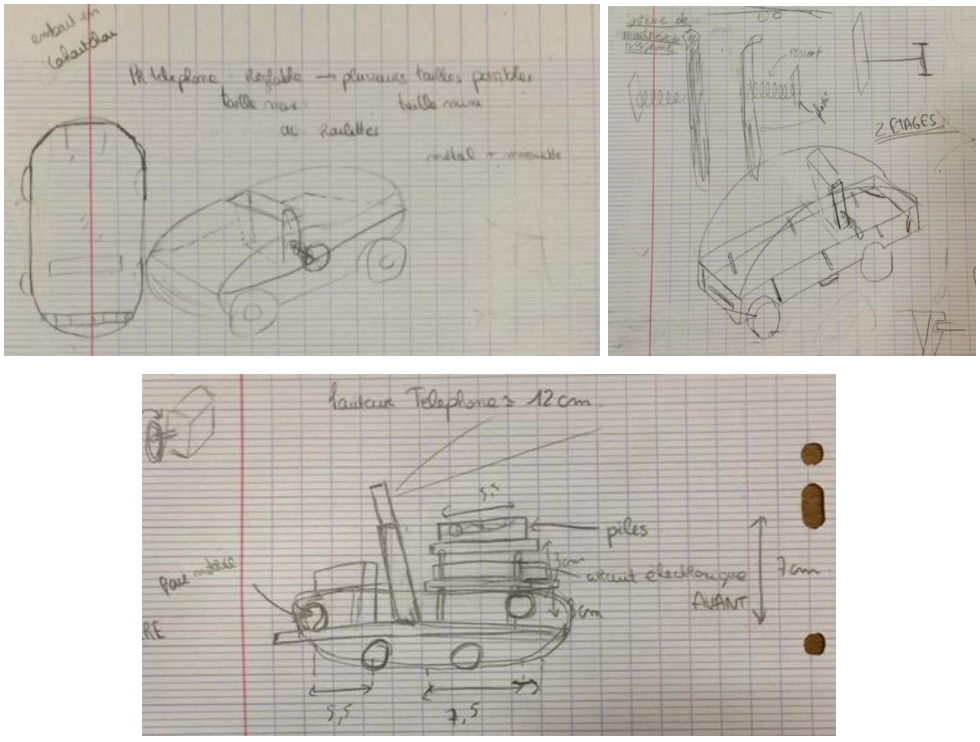
3.1. Partie mécanique

L'objectif de cette partie était de concevoir la forme du robot, en prenant en compte les matériaux à utiliser, puis de le modéliser en CAO et enfin de réaliser l'assemblage de toutes les pièces et composants nécessaires. En dernier lieu, il fallut réfléchir à la dimension esthétique du robot afin de le rendre plus attrayant.

La majeure partie a été faite constamment à deux, excepté la modélisation sur solidworks qui a été faite avant tout par Cécilia et les impressions 3D par Juliette.

3.1.1. Conception de la forme générale du robot

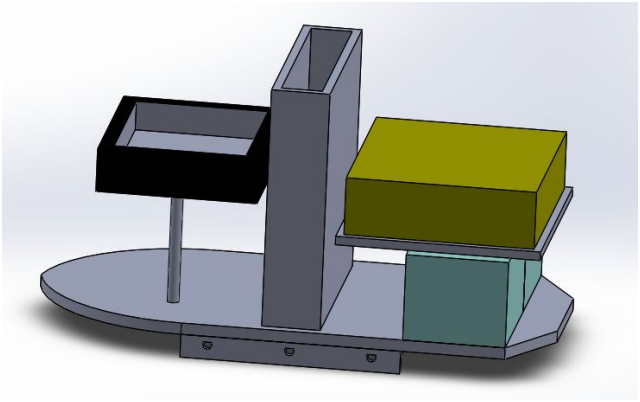
Avant de commencer la modélisation du robot à l'aide du logiciel de CAO Solidworks, nous avons réfléchi à la forme générale de notre robot ainsi qu'aux matériaux à utiliser. Nous voulions une forme originale et pas trop complexe afin qu'elle soit facilement réalisable en CAO. Après réflexion, quelques recherches sur internet et quelques modifications, nous nous sommes entendues sur la base d'un robot à deux étages, afin d'avoir assez d'espace pour placer les différents composants, tout en minimisant la taille du robot et sa hauteur pour des questions d'équilibre. Le robot est composé d'un châssis en bois (matériau récupéré), de roues avec chenilles, ainsi que d'un capot de forme arrondie, comportant un trou permettant l'insertion d'un téléphone portable. Il a fallu penser à la manière dont serait maintenu le téléphone servant à filmer et à piloter le robot dans le capot. Pour cela, nous avions premièrement pensé à un maintien grâce à un système de ressorts, cependant, face à la complexité de l'élaboration de ce système sur le robot, nous avons finalement décidé d'utiliser une petite plaque à visser sur le châssis et servant de butée. L'utilisation de chenilles au lieu de roues a pour d'obtenir un meilleur équilibre et une meilleure stabilité du robot en état de marche, notamment face aux petits obstacles pouvant être rencontrés. Concernant, les matériaux utilisés, nous avons tenté d'utiliser un maximum d'objets récupérés afin d'éviter le gaspillage. Avant de commencer la modélisation sur Solidworks, nous avons déterminé précisément les tailles de tous les éléments mécaniques et électroniques afin de déterminer la taille du châssis et de savoir comment agencer chaque éléments dessus.



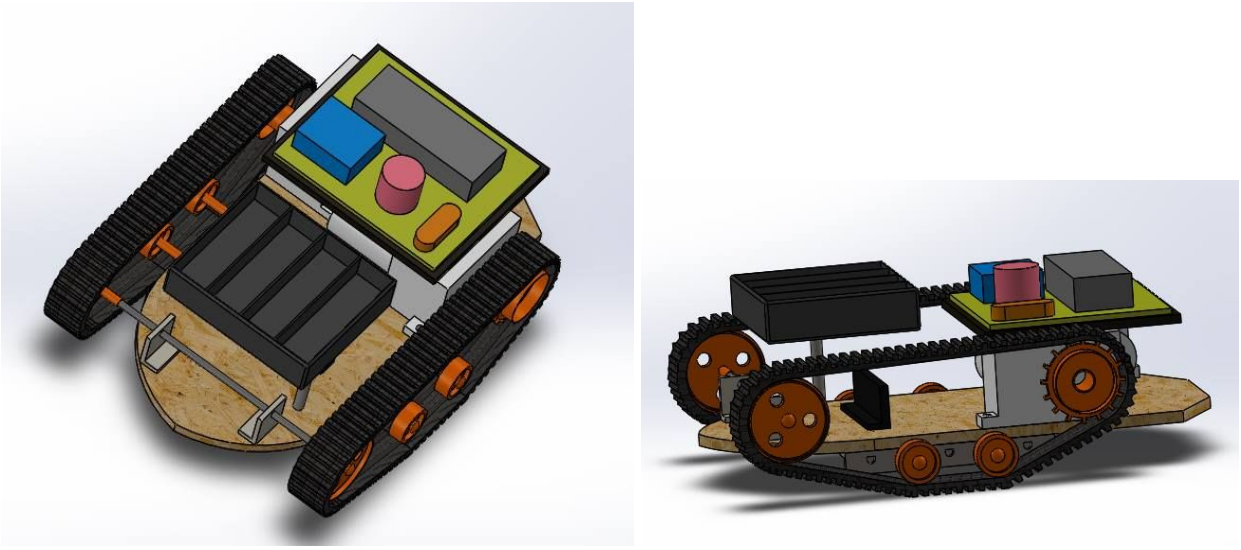
Premières esquisses du robot

3.1.2. Modélisation sur SolidWorks

La première pièce modélisée sous Solidworks fut le châssis, car il constitue la base de notre robot. Étant débutantes en CAO lorsque nous avons commencé la modélisation, nous n'étions pas totalement à l'aise avec le logiciel Solidworks. Nous avons donc dans un premier temps modélisé les différentes pièces du robot (hormis le châssis) par des blocs de dimensions correspondantes aux données relevées afin d'avoir un premier aperçu "semi-réaliste". Dans un second temps nous avons affiné cette modélisation pour avoir un rendu plus proche du réel. L'outil "apparence" de Solidworks, permettant de mettre de la couleur sur les pièces et d'en modifier la texture nous a permis d'avoir un rendu très réaliste de l'apparence de notre robot.



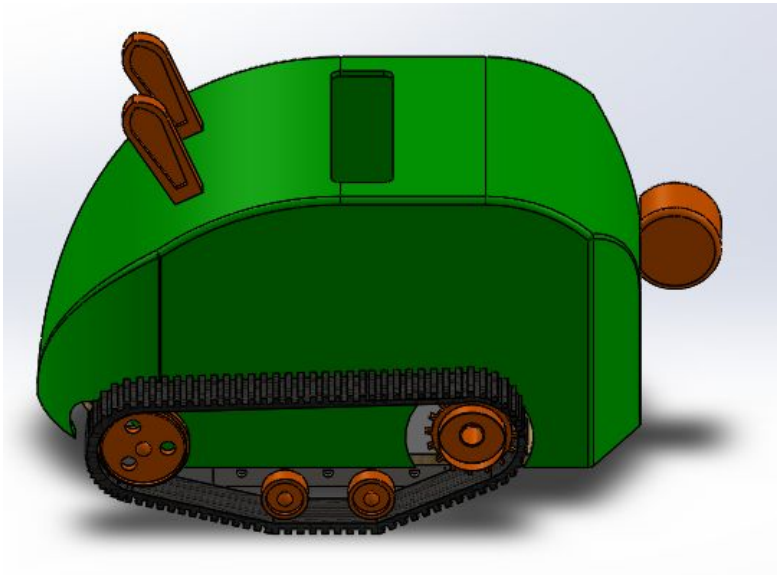
Modélisation par "blocs"



Modélisation du robot avec rendu réaliste

Lors de l'assemblage sur Solidworks, nous avons rencontrés des problèmes de dimensions sur certaines pièces (notamment les moteurs), qu'il nous a fallu redimensionner sur notre modèle. Nous avons également dû revoir à plusieurs reprises l'agencement des pièces sur le châssis et l'étage supérieur afin d'optimiser l'espace disponible. Cet assemblage sous solidworks nous a permis de savoir exactement ce qui pourrait nous poser problème lors du montage final et également de connaître exactement les endroits ou nous devions faire les trous dans le châssis pour poser les vis tenant les différentes pièces du robot.

En dernier lieu nous avons modélisé le capot, dont il a fallu réajuster les dimensions après le montage réel de toutes les pièces afin qu'il soit bien en place sur le robot.



Rendu final du robot

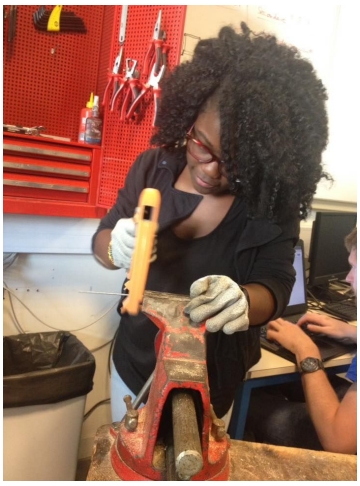
3.1.3. Fabrication des pièces et montage du robot

Une fois notre robot modélisé sur Solidworks, il fallu le monter:

Réalisation des trous dans le châssis

- Châssis : Pour la réalisation du châssis, nous avons commencé par récupérer une planche en bois dans le laboratoire de mécanique. Après avoir tracé au crayon à papier le contour du châssis, nous avons utilisé une scie pour l'usiner. Par la suite nous avons positionné les trous accueillant les vis afin de visser les plaques maintenant les axes de roues sous le châssis ainsi que les moteurs.

- Axes de roues : Pour les axes de roue, nous avons rencontré un problème. En effet, les axes à disposition au laboratoire étaient trop petits pour notre châssis. Il nous fallait donc trouver une solution de remplacement, un objet semblable avec le bon diamètre. Après avoir tenté différentes solutions (stylo, bâtonnets de brochettes, tiges de fouet...), nous avons finalement récupéré chez nous des électrodes en acier, dont nous avons enlevé l'enveloppe chimique à l'aide d'une pince, afin de ne garder que la partie métallique qui correspondait exactement aux diamètres des axes de roues mis à disposition. Nous les avons sciés afin d'obtenir des axes de la bonne longueur pour notre assemblage. Pour maintenir au mieux les axes dans les roues nous avons mis un peu de colle ultra forte dans les pinces maintenant les roues et ainsi empêché le petit jeu des axes présent.



Confection des axes de roues

- Montage différentes pièces : pour les deux moteurs, les plaques tenant les roues, le boitier pour les piles, la plaque maintenant le circuit électronique, l'axe de fer supportant le boitier, les roues et les chaînes nous les avons récupéré dans des pièces de robot électronique non utilisées mis à disposition par le professeur et nous les avons montés suivant notre assemblage solidworks.

- Boîtier à piles : nous l'avons surélevé car avec l'axe des roues avant, il ne tenait pas sur le châssis et nous cherchions le plus à optimiser l'espace du robot. Nous avons donc pensé à utiliser un pilier pour surélever le boitier de quelques millimètres.

- Placement du téléphone: nous avons pensé à faire une boite pour le téléphone mais il était plus intéressant pour optimiser l'espace de juste caler le téléphone entre les différentes pièces. Pour ce faire nous l'avons disposé entre les moteurs (l'étage au dessus des moteurs étant le circuit électronique), le boitier avec des petites cales récupérées sur un ancien robot et le capot ayant une ouverture conçue spécialement pour le smartphone. De cette manière le téléphone ne bouge pas lors de la mise en marche du robot.

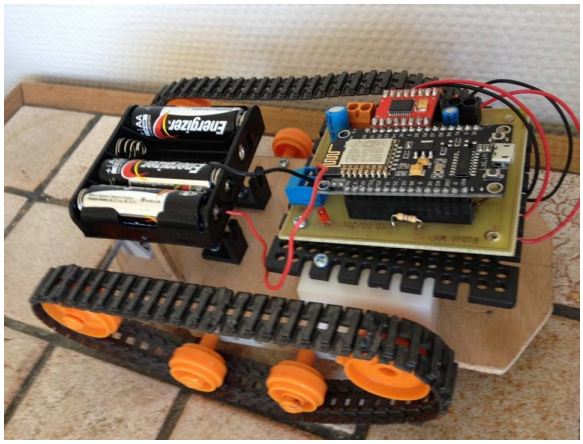
- Chaînes : nous avons récupéré des chaînes d'anciens robots mais n'ayant toujours pas les mêmes dimensions nous avons dû les couper et les rassembler, de ce fait, un petit jeu lors de la mise en route des chaînes est présent.

- Capot : nous avons trouvé intéressant d'exploiter l'imprimante 3D mise à disposition pour réaliser le capot de la forme souhaité. Après avoir réalisé le capot sous solidworks nous l'avons enregistré en fichier STL pour pouvoir le récupérer sous CURA, le logiciel pour l'imprimante Ultimaker 2+ et nous avons pu imprimer en mettant le fichier sur une carte SD.
Nous avons séparé le capot en 5 parties pour une meilleure impression et nous avons enlevé le surplus en le limant puis assemblé les parties. Nous avons pu rencontrer quelques problèmes lors de l'impression du fait du design de la pièce difficile à réaliser et de la première matière choisis (du PLA gris) qui posait beaucoup de dysfonctionnement de la machine. Avec l'aide du technicien, nous avons pu régler très vite ce problème et imprimer nos pièces. Le capot fini, il peut se disposer par le dessus du robot et être enlevé plus facilement pour regarder l'intérieur.

Le plus gros problème du montage a été que, pour les faire correspondre avec la partie électronique, il a souvent fallu démonter des pièces et les remonter au dernier moment par exemple, pour bien placer les fils qui se détachaient.

Nous notons également que le robot réel diffère très sensiblement de la modélisation effectuée sur Solidworks, car il a fallu qu'on s'adapte aux difficultés rencontrées.

Montage final



3.1.4. Dimension esthétique

Il nous semblait important de rendre le robot le plus esthétique possible, une partie souvent mise à l'écart. De ce fait, nous avons adouci les formes et les avons rendus plus design, tel que le châssis avec une partie arrondie (le devant) et des chanfreins (arrière). Cette dimension esthétique est l'une des raisons pour lesquelles nous avons préféré travailler sur un châssis en bois sur lequel nous pouvions changer facilement la forme.

La majeure partie de l'esthétique s'est joué sur le capot. Nous ne pouvions faire une forme vraiment complexe, c'est pourquoi nous avons décidé d'ajouter une touche originale avec un capot en forme de lapin. (Le peindre nous a également permi de le rendre encore plus agréable et original à vue d'oeil).



Rendu final du robot RSP6

3.2. Application Android et carte NodeMCU V3.

Grâce à un camarade d'ASI qui avait déjà réalisé une application Android en classe de Terminal, Damien a appris qu'il était possible de réaliser une application sans savoir coder en Java grâce à MIT App Inventor, un logiciel en ligne. Après avoir regardé quelques tutoriels avec Antoine, nous en avons trouvé un qui a permis de réellement commencer l'application pour le projet. Il s'agissait d'entrer l'URL d'un site dans l'application et la page du site s'affichait également dans l'application. C'est exactement ce qu'il fallait pour afficher la vidéo du portable sur le robot avec la page html générée par la NodeMCU V3. En effet, Guillaume et Corentin ont trouvé une application sur le Google Play Store qui s'appelle IP Webcam. Cette application permet entre autres qu'un smartphone prenne le contrôle de la caméra d'un second téléphone à travers la WIFI (vidéo, flash, zoom...).

Maintenant que nous avions une base pour la partie informatique, Antoine et Guillaume se sont penchés sur la partie électronique tandis que Corentin et Damien ont poursuivi la partie informatique.

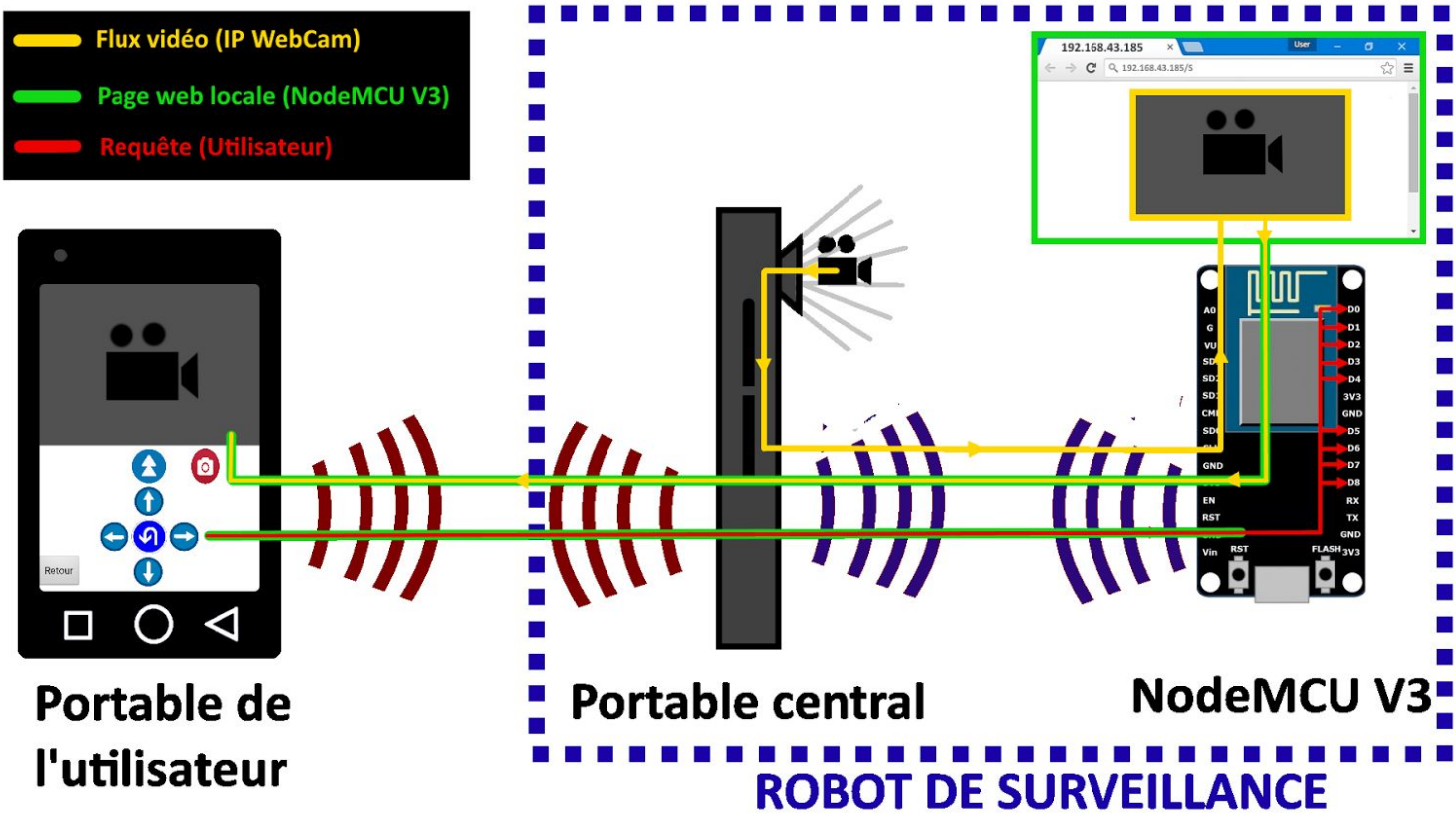
Avant d'étudier dans le détail les codes de l'application et le code arduino, intéressons-nous au fonctionnement général du robot.

Tout d'abord le portable de l'utilisateur et la NodeMCU V3 doivent être connectés à un portable central qui se trouve sur le robot. Ils communiqueront par son intermédiaire. Ce portable crée un réseau local (il faut activer la fonction « Hotspot ») sur lequel l'utilisateur se connectera comme à un réseau Wifi classique. Le SSID, c'est-à-dire l'adresse du réseau et le mot de passe associé ont été enregistrés dans le code de la NodeMCU V3 sous la forme de constantes. Celle-ci peut donc également se connecter sur le portable central et ce, automatiquement.

Une fois les connections établis, l'utilisateur est prêt à utiliser le robot par le biais d'une application dont le fonctionnement sera expliqué par la suite. (Cf 3.2.1).

Voyons maintenant le fonctionnement des communications entres les différents appareils connectés au réseau :

Schéma du principe de fonctionnement du robot de sécurité :



Le flux vidéo capté par l'application IP Webcam (installée sur le portable central) est récupéré par la carte NodeMCU V3 qui va l'intégrer à une page web disponible sur le réseau local. Cette page web va ensuite être transmise à l'application RSP6 afin de permettre à l'utilisateur de voir ce qu'il se passe devant le robot.

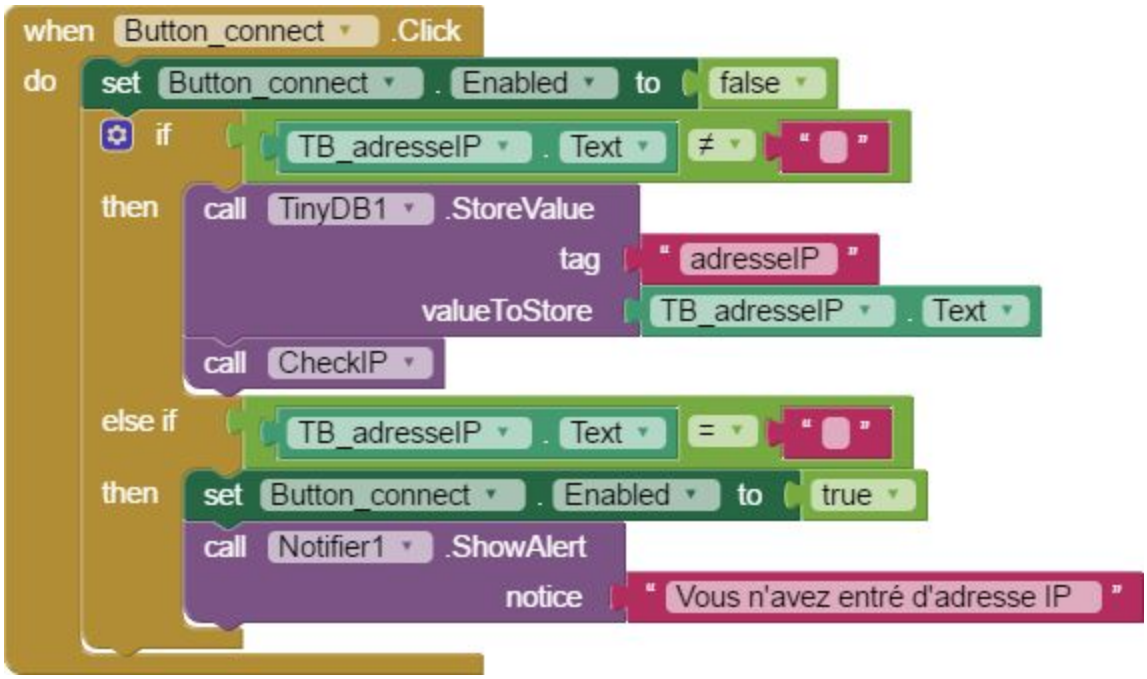
Ensuite, lorsque l'utilisateur appuie sur un bouton, une requête est envoyée à la NodeMCU V3 (par exemple : 192.168.43.185/exemple). En soit l'utilisateur ne fait que demander l'accès à la page « exemple » du site créée par la carte. Comme il aurait pu demander l'accès à wikipedia à la page arduino en tapant « https://fr.wikipedia.org/Arduino ». Sauf que notre carte ne va pas traiter cette demande de la même manière que le serveur de wikipedia, en fait elle va renvoyer la même page quelle que soit la requête (sauf dans le cas où l'utilisateur souhaite prendre une photo de ce que voit le robot), cependant elle va activer ou désactiver le courant dans certaines de ses sorties. Par exemple si la requête est 192.168.43.185/S, la carte comprend le /S comme un « stop » et nous l'avons programmé pour qu'elle coupe les moteurs dans ce cas précis.

3.2.1. Code MIT App Inventor

Nous avons décidé de nommer l'application "RSP6" (Robot Sécurité P6)

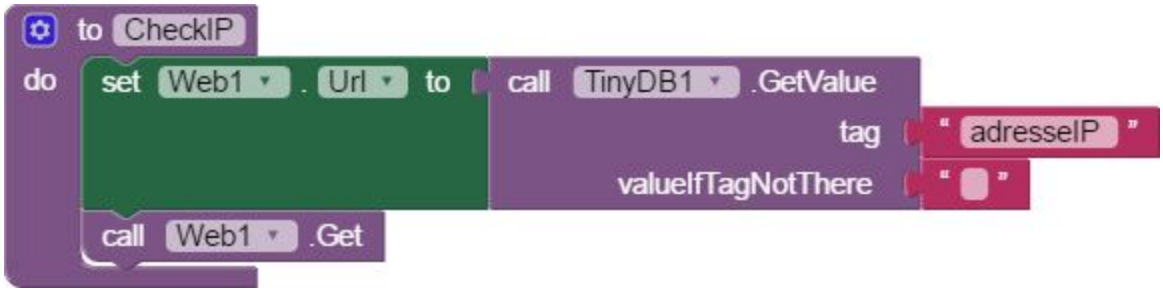
L'application fonctionne sur trois écrans :
- écran 1 : Dès que l'application est lancée, si aucun dossier n'a été créé, l'application crée un dossier "Photos_RSP6" où des photos prises à travers l'application seront sauvegardées. L'utilisateur entre l'adresse IP de la carte NodeMCU V3 et clique sur un bouton pour se connecter.
- écran 2 : dans la partie supérieur de cet écran, l'utilisateur voit ce que le portable sur le robot voit. Dans la partie inférieur, l'utilisateur peut contrôler : aller tout droit rapidement, tout droit normalement, tourner à droite, tourner à gauche, reculer, faire un demi-tour. Ces cinq boutons fonctionnent de la manière suivante : lorsque l'utilisateur reste appuyé sur le bouton, le robot effectue l'action demandée et lorsque l'utilisateur relâche le bouton, le robot s'arrête. Dans cette même partie de l'écran, l'utilisateur à la possibilité de prendre une photo de ce que le portable sur le robot voit, c'est à dire prendre un screenshot de ce qui est présent dans la partie de l'écran 2. En effet, comme nous ne savions pas comment activer la caméra du portable sur le robot à partir du portable que l'on a dans la main, nous avons décidé de prendre un screenshot de la vidéo sur le portable dans la main, ce qui revient à prendre une photo à partir du portable sur le robot.
- écran 3 : la partie supérieur de l'écran 2 prend maintenant quasiment toute la place de l'écran 3. En bas de l'écran 3, il y a un bouton sur lequel l'utilisateur peut cliquer pour confirmer qu'il veut prendre une photo (un screenshot en réalité). Quand l'utilisateur clique dessus, le bouton disparaît permettant à la partie supérieur de l'écran 2 de prendre l'intégralité de l'écran 3. Ce screenshot est sauvegardé dans un dossier créé par l'application dans l'écran 1. L'utilisateur peut retrouver ces photos dans la galerie.

3.2.1.1. Ecran 1

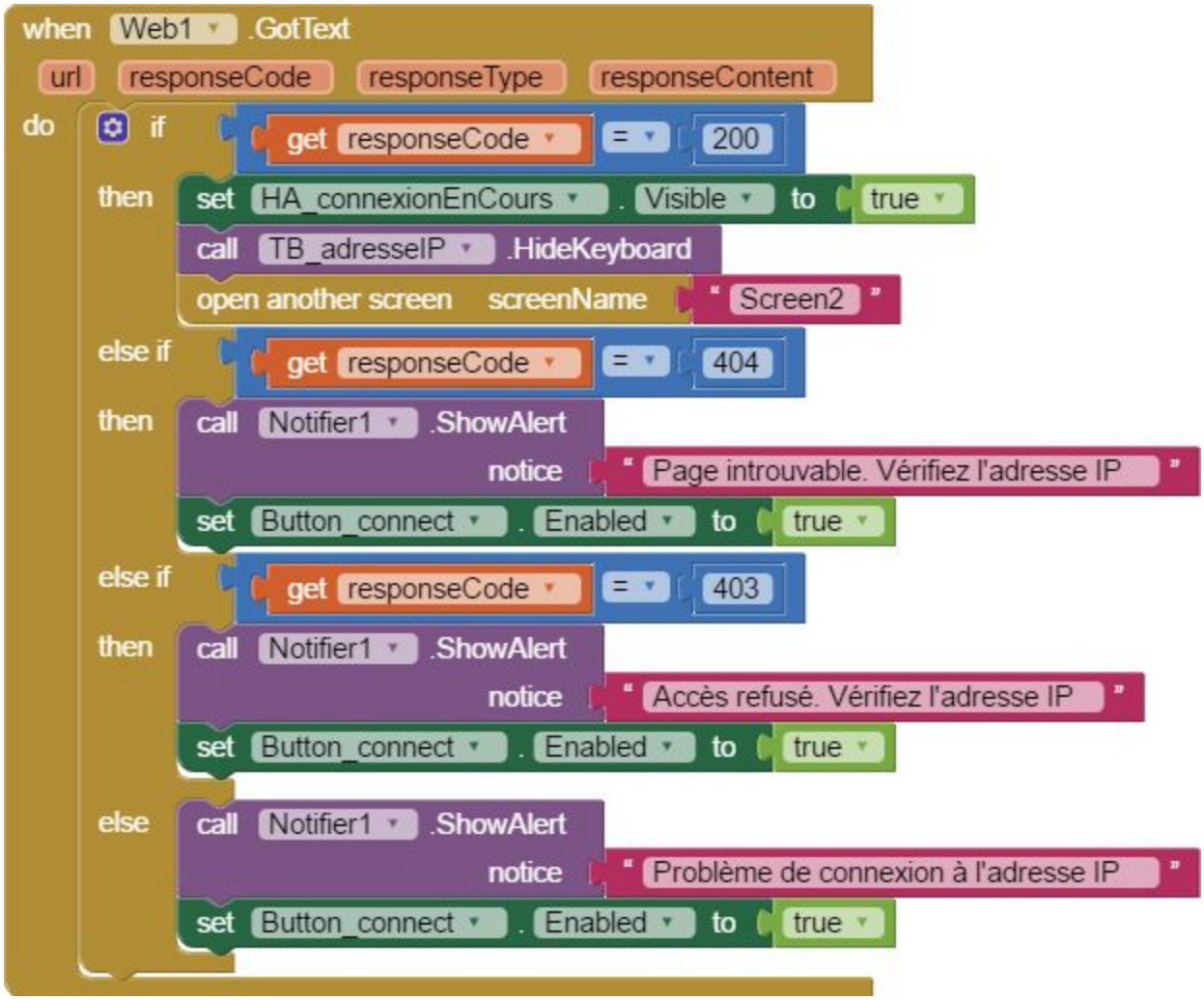


Lorsque le bouton “Tap to Connect” 

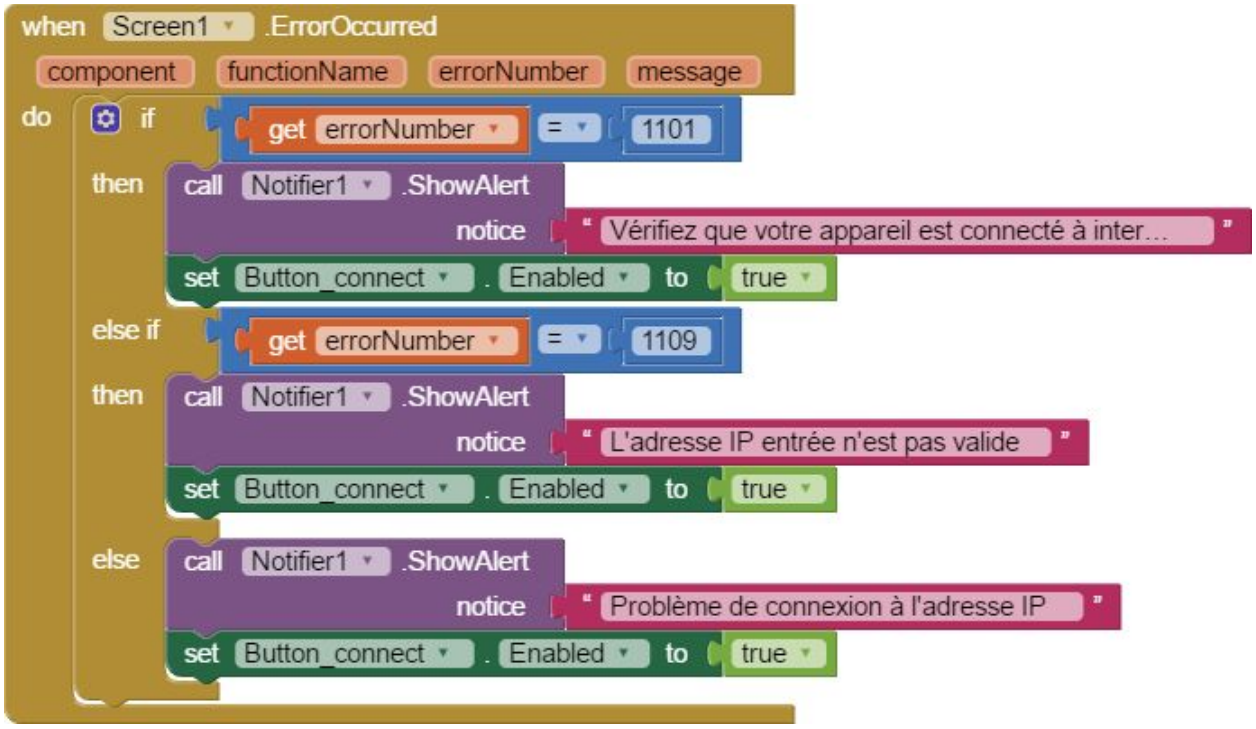
est appuyé, le bouton est rendu inutilisable. Si une adresse IP a été entrée alors on vérifie la validité de l’adresse entrée sinon, si aucune adresse IP n’a été entrée, un message d’alerte apparaît pour dire à l’utilisateur qu’aucune adresse IP n’a été entrée. Rendre le bouton “Tap to Connect” inutilisable temporairement est nécessaire car si l’adresse IP entrée est valide et l’utilisateur clique plusieurs fois à la suite sur “Tap to Connect”, plusieurs écrans 2 s’ouvrent.



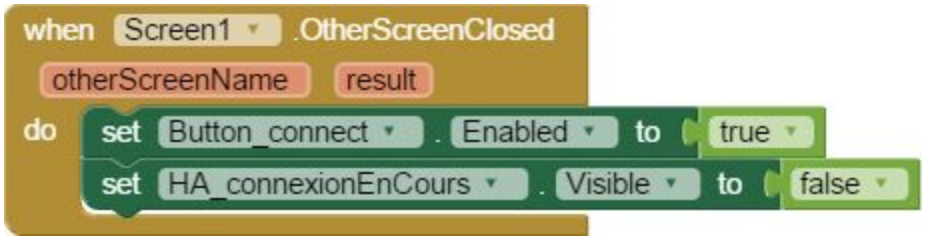
Cette procédure CheckIP vérifie la validité de l’adresse IP entrée.



On interprète la réponse de la procédure “call Web1.Get”. Si cette réponse est égale à 200 alors la connexion avec la carte NodeMCU V3 a été réalisée avec succès donc l’adresse IP est valide. Dans ce cas, on rend le message “Connexion en cours…” visible sachant que de base il est invisible puis on cache le clavier. Si la réponse est égale à 404 alors la page est introuvable et un message d’alerte adéquat s’affiche et on rend le bouton “Tap to Connect” réutilisable. Un autre message d’alerte s’affiche si la réponse est égale à 403 et de même, on rend le bouton “Tap to Connect” réutilisable. Si la réponse est autre, on dit à l’utilisateur qu’il y a un problème de connexion et rend le bouton “Tap to Connect” réutilisable.



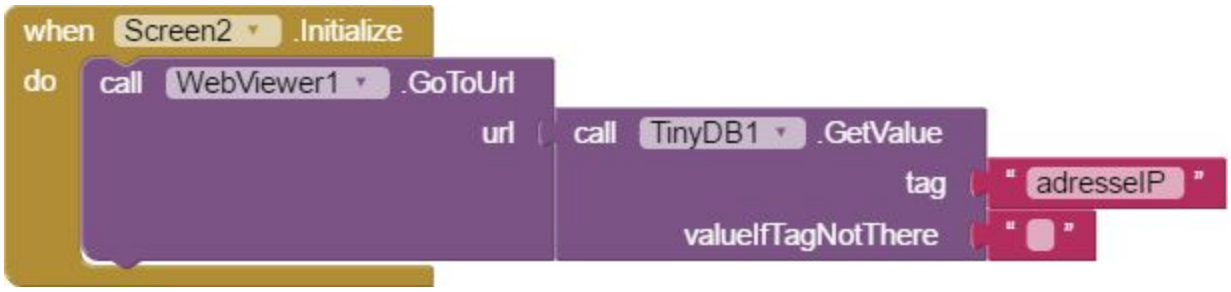
Si une erreur se produit dans l'application lors de la procédure "call Web1.Get", différents messages d'alertes s'affichent en fonction de l'erreur. Dans chaque cas, rend le bouton "Tap to Connect" réutilisable.



Lorsque l'écran 2 est fermé, on rend le bouton "Tap to Connect" réutilisable et on rend le message "Connexion en cours…" invisible.

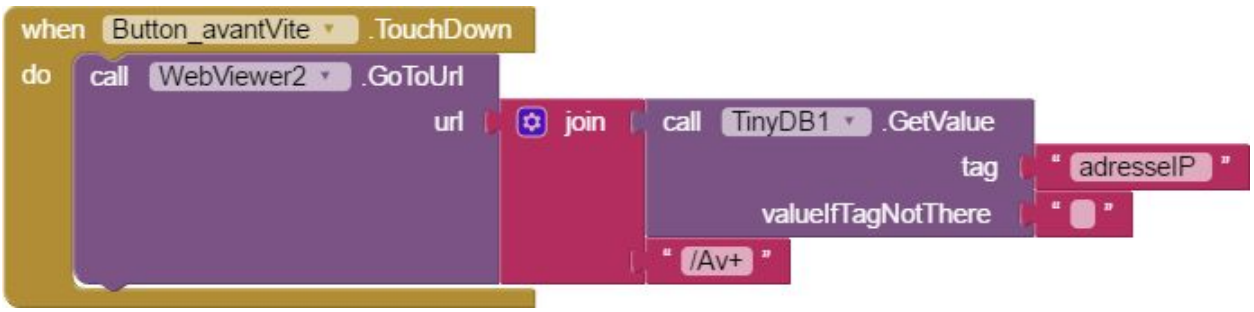
Lorsque le bouton "Effacer" Effacer est appuyé, le contenu du textbox pour l'adresse IP est vidé.

3.2.1.2. Ecran 2

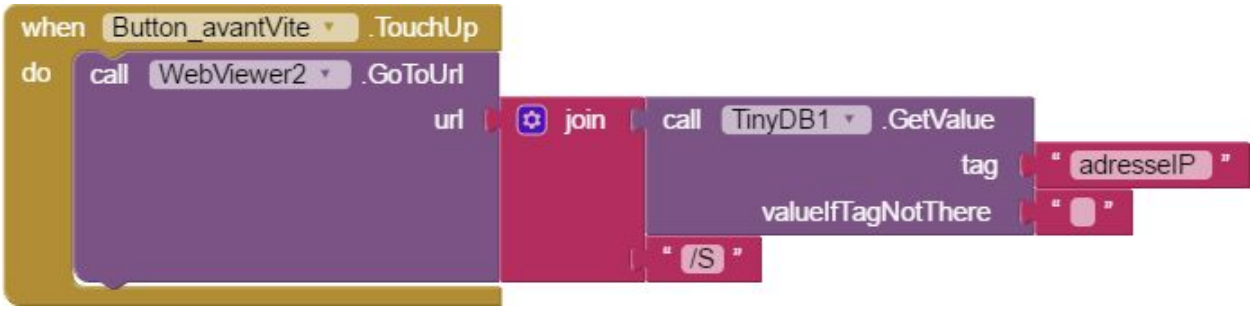


Lorsque l'écran 2 s'initialise, on se connecte à l'adresse IP et on affiche la page html de la carte NodeMCU V3 dans la partie supérieur de l'écran 2. Cette partie supérieure contient le contenu de la caméra du portable sur le robot.

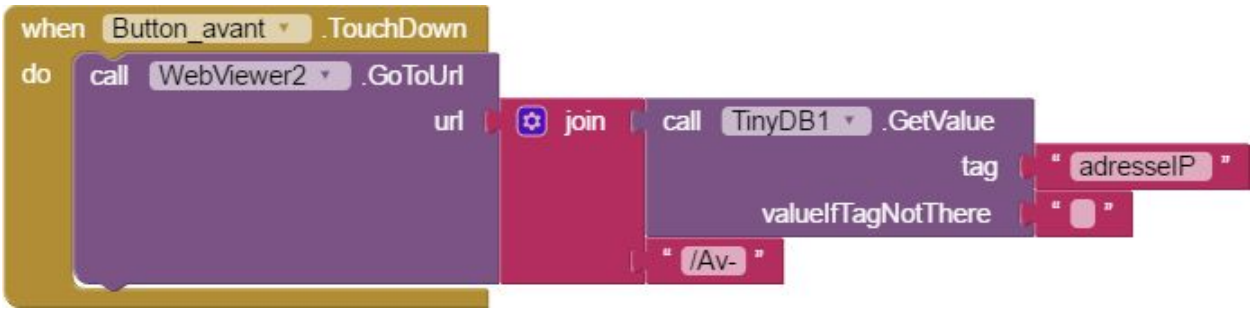
Lorsque le bouton "Retour" Retour est appuyé, on l'écran 2 est fermé et on retourne donc à l'écran 1.



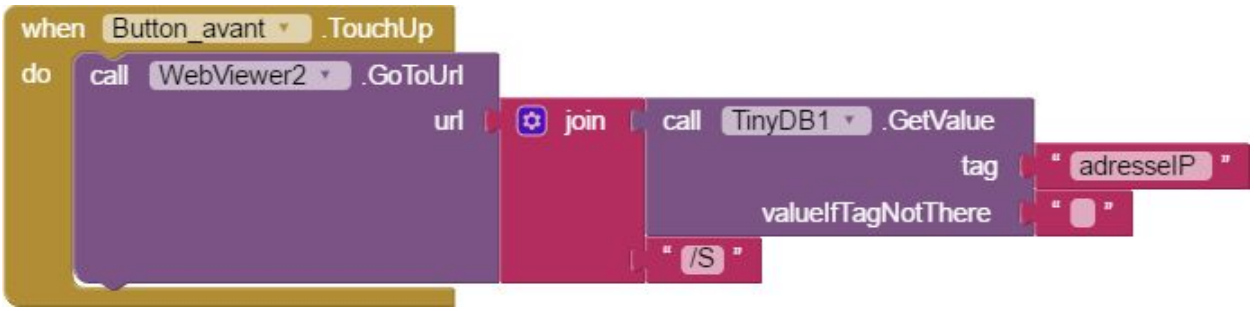
Lorsque l'utilisateur reste appuyé sur le bouton , le robot avance rapidement.



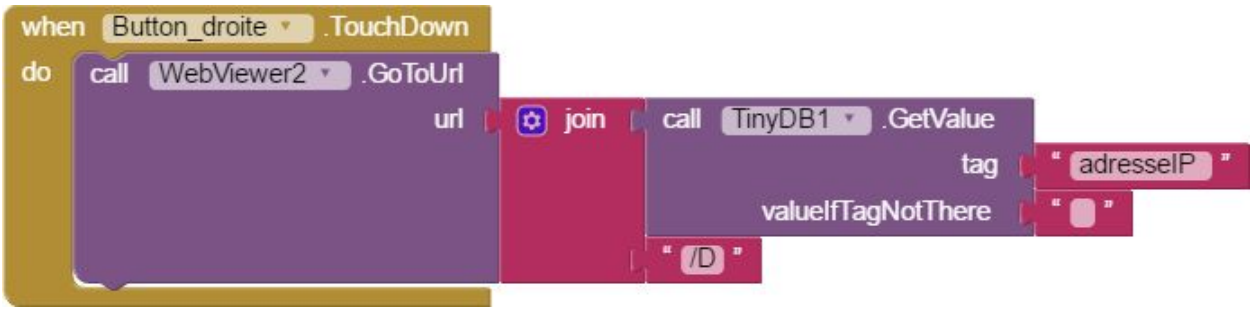
Lorsque l'utilisateur relâche le bouton , le robot s'arrête.



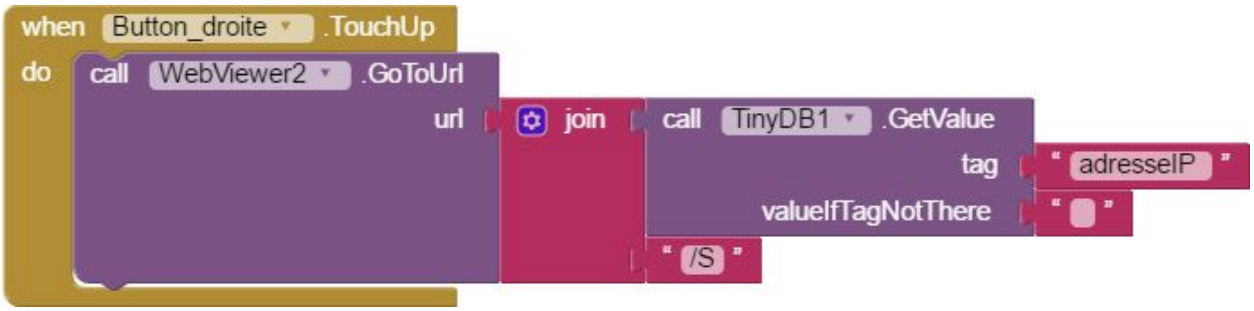
Lorsque l'utilisateur reste appuyé sur le bouton , le robot avance normalement.



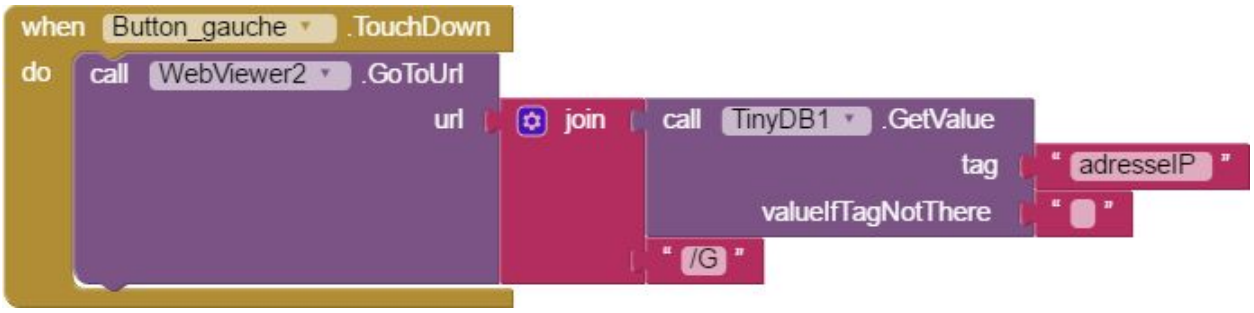
Lorsque l'utilisateur relâche le bouton , le robot s'arrête.



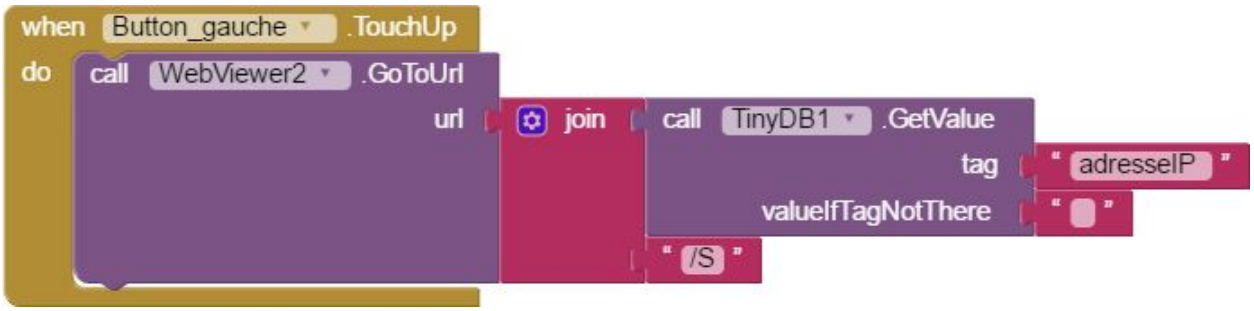
Lorsque l'utilisateur reste appuyé sur le bouton , le robot tourne à droite.



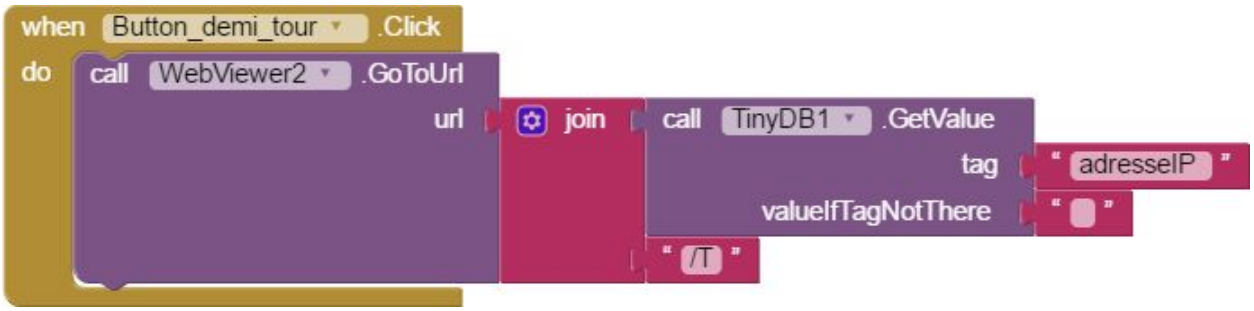
Lorsque l'utilisateur relâche le bouton , le robot s'arrête.



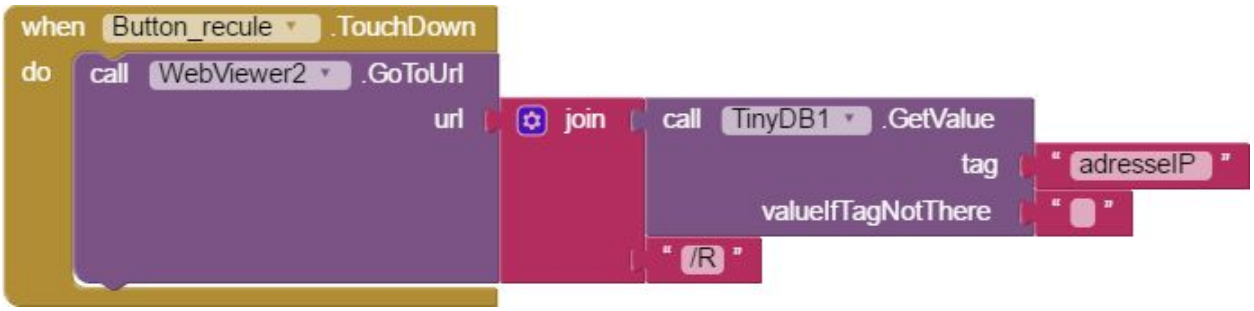
Lorsque l'utilisateur reste appuyé sur le bouton , le robot tourne à gauche.



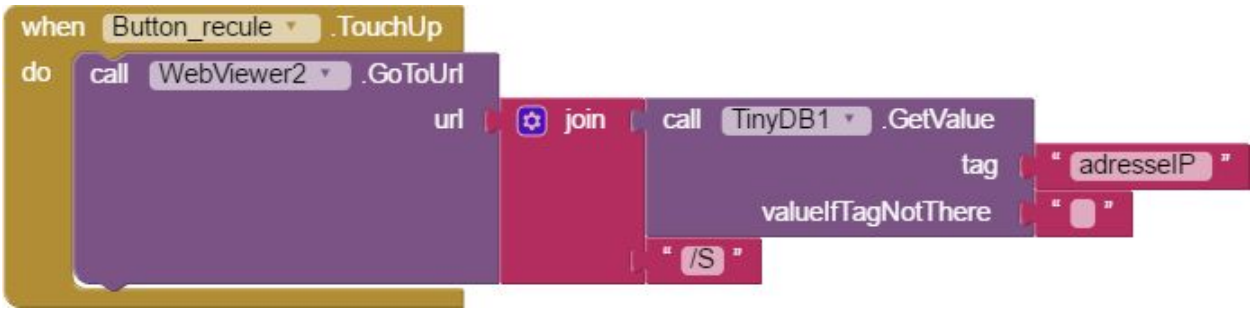
Lorsque l'utilisateur relâche le bouton , le robot s'arrête.



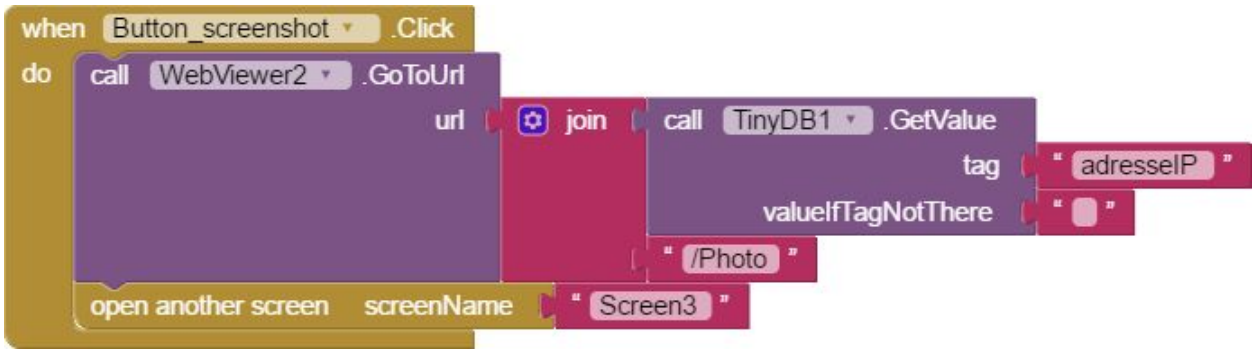
Lorsque l'utilisateur clique sur le bouton , le robot effectue un demi-tour.



Lorsque l'utilisateur reste appuyé sur le bouton , le robot recule.

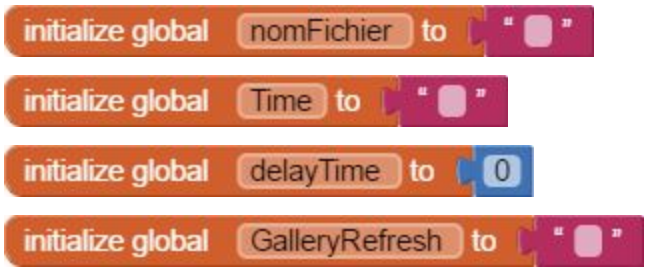


Lorsque l'utilisateur relâche le bouton , le robot s'arrête.

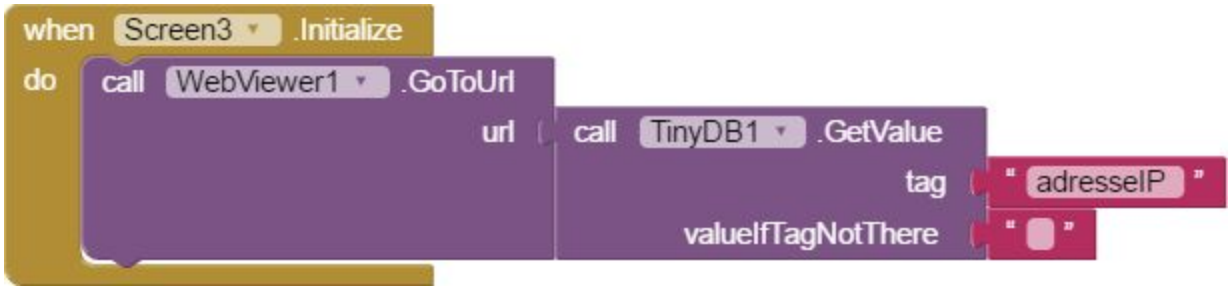


Lorsque l'utilisateur clique sur le bouton , les dimensions de la vidéo sur la page html de la carte NodeMCU V3 sont modifiées et l'application ouvre l'écran 3.

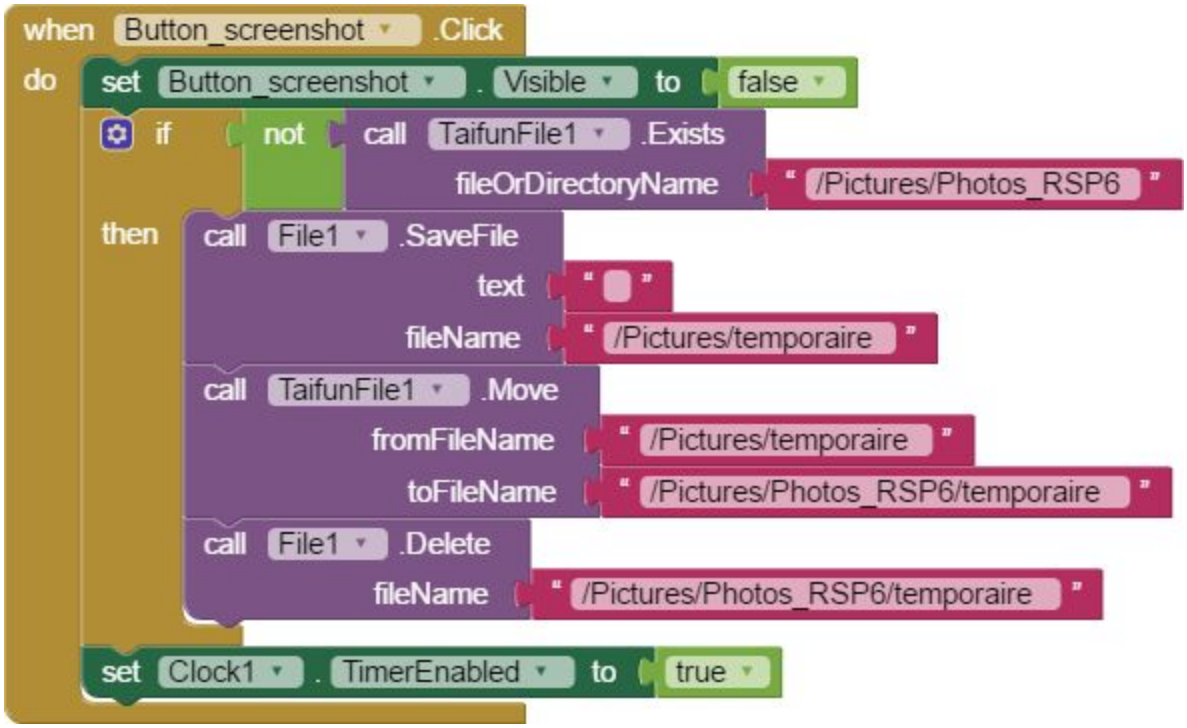
3.2.1.3. Ecran 3



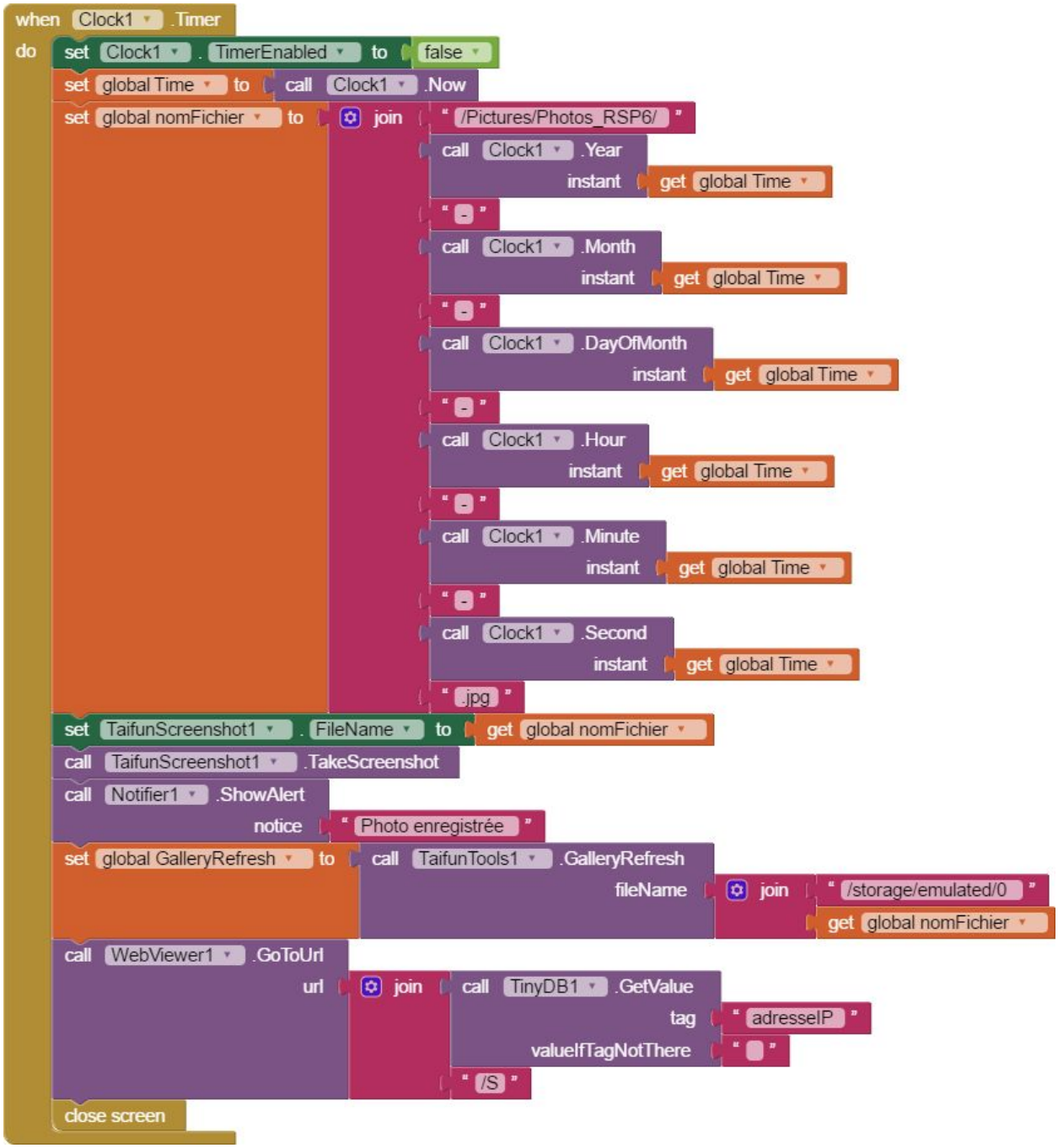
Quand l'écran 3 s'ouvre, on initialise des variables globales. Trois variables sont du texte (caractère ou chaîne de caractères) et une variable est un nombre (entier ou réel).



Lorsque l'écran 3 s'initialise, on se connecte à l'adresse IP et on affiche la page html de la carte NodeMCU V3.



Lorsque l'utilisateur clique sur le bouton Clickez ici pour prendre la photo, ce bouton est rendu invisible. S'il n'existe pas encore, on créer un nouveau répertoire qui s'appelle "Photos_RSP6" dans le répertoire "Pictures" dont le chemin est "/storage/emulated/0/Pictures". Ensuite un délai d'une seconde commence. Comme MIT App Inventor est en mode beta, toutes les fonctionnalités ne sont pas disponibles. Par exemple, il n'y a pas la possibilité de créer un répertoire directement. Le moyen que nous avons trouvé pour le faire nécessite de créer un fichier temporaire dans un répertoire existant et de le déplacer dans un répertoire qui n'existe pas encore. Quand ce répertoire n'existe pas, il est créé puis on supprime le fichier temporaire présent le nouveau répertoire créé.



Après ce délai d'une seconde, on crée un nom de fichier pour la photo qui va être prise. Pour être sûr que le nom de fichier soit nouveau à chaque fois, le nom est de la forme suivante : "Année-Mois-Jour-Heure-Minute-Seconde". Ensuite le screenshot est effectué et un message d'alerte apparaît pour informer l'utilisateur que la photo a bien été enregistrée. Chaque est photo est sauvegardée dans le répertoire "Photos_RSP6" qui se trouve dans le répertoire Pictures.

Pour que les photos apparaissent dans la galerie automatiquement, c'est à dire que l'utilisateur n'ait pas besoin de redémarrer son appareil, on utilise la fonction "call TaifunTools1.GalleryRefresh" qui retourne le chemin

“/storage/emulated/0/Pictures/Photos_RPSP6/Année-Mois-Jour-Heure-Minute-Seconde”. Cette fonction permet donc de rendre la photo qui vient d’être prise accessible dans la galerie.

Pour revenir aux dimensions initiales de la vidéo sur page html (adaptées à l’écran 2), on arrête le robot (même s’il était déjà l’arrêt).

Enfin, l’écran 3 est fermé.

3.2.2.1 Code Arduino - méthode de développement

La carte NodeMCU V3 fonctionne de la même manière qu’un arduino, avec la possibilité d’utiliser un réseau wifi en plus. Pour la programmer nous avons donc utilisé l’IDE Arduino en version 1.8.3 comme nous l’aurions fait pour une carte arduino classique. Il a juste été nécessaire de paramétrer correctement l’IDE pour qu’il fonctionne avec notre carte.

Comme nous n’avions aucune connaissance concernant le C++ (langage utilisé pour coder les arduinos), Monsieur DELAMARE nous a fourni un code permettant de contrôler un robot par wifi afin de nous en inspirer. Nous avons gardé le principe de contrôle du robot via des requêtes serveur comme sur son modèle mais avons dû changer bien des choses, notamment afin de rajouter la vidéo sur la page web générée par le robot. Le code étant assez répétitif par moment, plutôt que de l’expliquer point par point ici, je vous recommande d’aller le voir en annexe. Il est entièrement commenté de manière à comprendre facilement son fonctionnement.

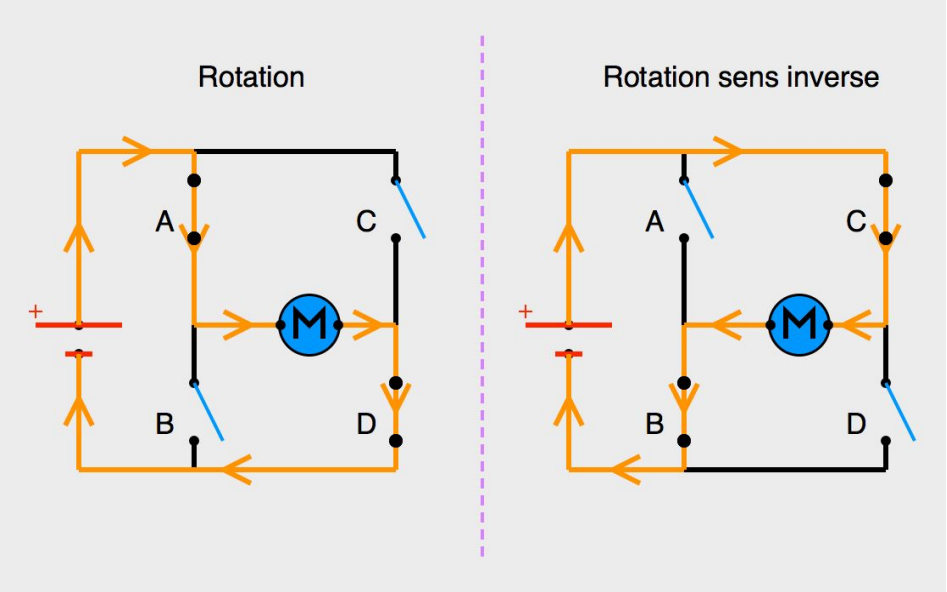
3.2.2.2 Problème rencontré

Le schéma du circuit imprimé fourni par monsieur DELAMARE ne correspondait pas à celui utilisé sur le prototype réalisé par ce dernier. Le programme arduino que nous avions développé sur le prototype n’était donc pas adapté à notre robot mais nous l’ignorions. Lors de la dernière séance du projet nous avions donc un robot qui ne répondait pas du tout comme il aurait dû. Diagnostiquer le problème n’a pas été évident d’autant plus que nous nous pensions au départ qu’il s’agissait d’un problème de soudure. Finalement après une longue analyse du schéma du circuit imprimé, nous nous sommes rendu compte qu’il y avait un problème de correspondance entre les sorties dans le programme et celles sur le circuit imprimé. Il a alors suffit de réadapter le programme à la nouvelle configuration.

3.3. Partie électronique

3.3.1 Travail réalisé

Lors de la première séance de ce projet, Monsieur DELAMARE nous a expliqué comment procéder pour contrôler les moteurs. La notion à retenir était le pont en H. En effet, on était à première vue capables de faire tourner un moteur dans un seul et unique sens. Pour le faire tourner dans l'autre sens, il fallait inverser les branchements ! Pour résoudre ce problème, nous avons vu qu'il existe la méthode du pont en H, permettant de contrôler les moteurs sans même changer un branchement.



Sens du courant en fonction de l'état des interrupteurs

Le principe est le suivant : les interrupteurs fonctionnent deux par deux. Le A est associé au D et le B est associé au C. Voyons ce qui arrive lorsqu'on actionne en même temps les interrupteurs A et D (schéma de gauche), ou les interrupteurs B et C (schéma de droite) :
- Sur le schéma de gauche : les interrupteurs A et D sont fermés. Le courant entre par la patte gauche du moteur et sort par sa droite. Le moteur tourne.
- Sur le schéma de droite : les interrupteurs B et C sont fermés. Le courant entre par la patte droite du moteur et sort par sa gauche. Le moteur tourne donc dans le sens inverse !

On préfère utiliser des transistors à la place des interrupteurs mais le principe reste le même. Il faut cependant ne pas oublier de les protéger du retour de tension. Pour cela, il suffit de placer des diodes au bons endroits.

Il existe des puces qui fonctionnent sur ce principe. Nous utilisons ici la TB6612FNG qui comporte deux ponts en H (un pour chaque moteur).

Dans un premier temps, nous nous sommes familiarisés avec le matériel mis à notre disposition et nous avons élaboré la liste des composants dont nous avions besoin pour la réalisation de notre projet. Puis nous nous sommes rendus compte que la partie électronique devait être laissée de côté pendant quelques séance au profit de la partie informatique car par exemple, il ne sert à rien d'alimenter les moteurs quand on n'a pas encore expliqué au robot comment il doit réagir lorsqu'on lui demande d'avancer. Nous avons donc mis cette partie de côté. Puis, vers la fin du semestre, lorsque l'application est devenue fonctionnelle et que la carte imprimée nous est parvenue, nous avons pu passé à la partie électronique. Une fois assurés de la bonne compatibilité des composants et du fonctionnement de nos programmes, nous nous sommes lancés dans la soudure afin de fixer les connections nécessaires entre nos différents composants. Voici la liste de ceux présents sur notre circuit imprimé :

- 1 résistance,
- 1 LED,
- 3 prises sur lesquelles seront connectés les deux moteurs ainsi que l'alimentation,
- 3 condensateurs,
- 4 barrettes de pins qui seront les supports de la NodeMCU V3 et du shield TB6612FNG permettant de contrôler les deux moteur.

La soudure, une grande découverte

Notre expérience en soudure étant presque inexistante, nous avons été très consciencieux lors de la soudure des petits composants (tout en sachant que nos soudures ne sont pas les meilleures, loin de là). Pour autant, cette découverte s'est très bien passée et nous a même donné goût aux soudures ! Nous avions accès à l'atelier de mécanique qui mettait à disposition des fers à souder, des pinces coupantes ainsi que notre meilleur ami le multimètre qui nous permettait de vérifier si les liaisons entre chaques points du circuit étaient bien réalisées.

3.3.2 Problèmes rencontrés

La précision était de mise car comme les différents chemins sur le circuit imprimé sont très proches les uns des autres, il a fallu faire très attention à ne pas les relier par mégarde lors de la soudure. Cela nous a demandé beaucoup de concentration mais également quelques retouches.

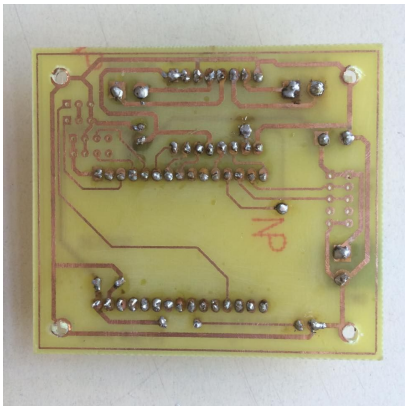
De plus, à cause d'une maladresse, nous nous sommes retrouvés à souder un composant au mauvais endroit. Mais nous avons habilement surmonté cette épreuve sans endommager la carte imprimée.

Aussi le circuit présentait un défaut : deux des chemins étaient coupés ce qui empêchait le courant de passer d'un point A à un point B. Pour résoudre ce problème, nous avons utilisé deux méthodes différentes. Ainsi, le premier chemin coupé était assez long et nous avons pu souder une patte de résistance pour permettre le passage du courant. Le second chemin étant très petit, nous avons dû procéder autrement. Nous avons utilisé une sorte de fil électrique très pratique pour relier deux point rapprochés. En effet, ce fil se dénude de lui-même au moment de la soudure nous permettant de gagner en précision.

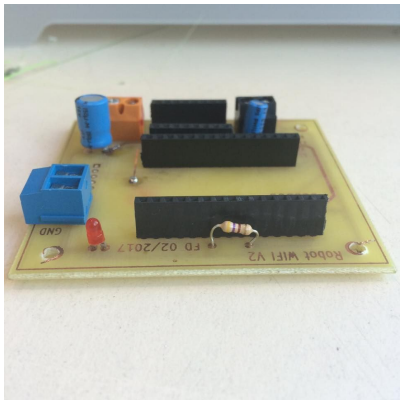
Lors du test final de notre robot, nous avons été surpris de voir qu'un problème subsistait. En effet, la LED ne s'allumait pas et nous devions brancher le robot à l'ordinateur en plus de l'alimentation fournie par les piles. Après un minutieux examen du circuit, nous nous sommes aperçus avec l'aide de notre professeur qu'une soudure manquait au niveau d'un condensateur, ce qui coupait la circulation du courant dans une petite partie du circuit. Après rectification, nous avons pu observer que notre robot roulait comme sur des… chenilles !

3.3.3 Résultat

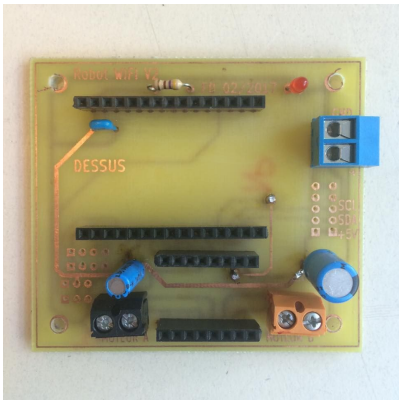
Finalement, voici un petit aperçu de notre travail.



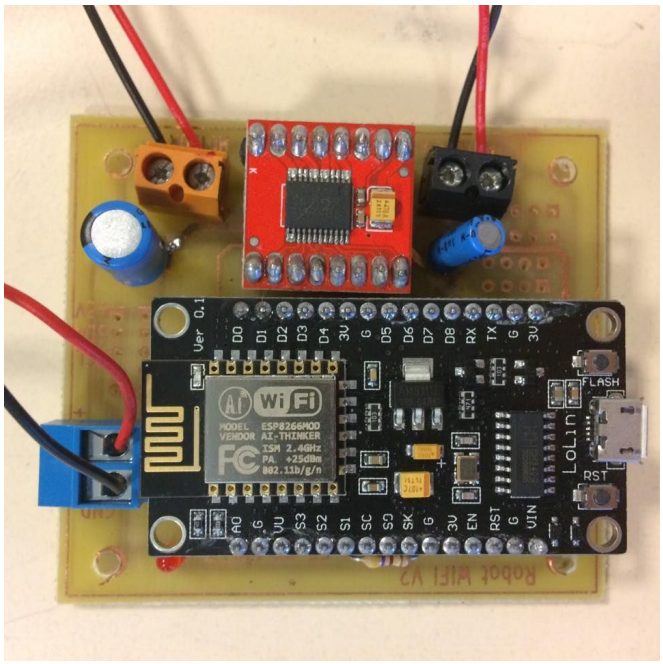
Carte vue de dessous



Carte vu de profil



Carte vue de dessus



Aperçu final

4. Conclusions et perspectives

Nous sommes parvenus à atteindre le but du projet, à savoir réaliser un robot mobile de surveillance piloté par WIFI à partir d'un smartphone. Nous avons pu réaliser toutes nos idées de départ, nous sommes donc très satisfaits du résultat.

Conclusions sur l'apport personnel de cet E.C. projet :

Antoine BAILLIOT : Ce projet a été pour moi un aperçu d'une application concrète de ce que l'on peut apprendre en cours et une découverte agréable de la robotique. C'est un projet qui m'a permis de renforcer mon organisation pour le travail en équipe, et dont l'enrichissement ne peut être négligé. En effet, étant au départ assigné à la partie informatique, cela m'a permis de développer mes connaissances en arduino, et avec mes recherches sur le sujet, de découvrir beaucoup d'applications possibles avec ce langage. Cependant, comme le code arduino a été par la suite donné par l'enseignant, Je me suis alors redirigé vers la partie électronique et j'ai ainsi pu découvrir la soudure et enrichir mes connaissances en électroniques. Finalement, je garderai de ce projet un souvenir agréable, la satisfaction d'avoir construit, avec l'aide de mes camarades, un robot de surveillance de nous-mêmes avec le matériel fourni.

Guillaume DRON : Pour moi, ce projet a été une excellente expérience. Outre le travail en équipe qui reste toujours un élément enrichissant dans un projet de groupe, j'ai pu m'initier à l'électronique, un domaine qui m'avait toujours attiré, tout en parfaisant mes connaissances en informatique. Ce projet m'a également permis de développer ma créativité, mon autonomie et ma réactivité face aux imprévus. Il m'a finalement permis de confirmer mon choix d'orientation pour l'année prochaine (ASI).

Cécilia MALONGA : Ce projet a été pour moi une expérience très enrichissante et plaisante, aussi bien humainement qu'intellectuellement. J'ai trouvé constructif le fait d'avoir à réfléchir en groupe, dans un but commun et sur un projet aussi concret et polyvalent, regroupant plusieurs domaines scientifiques à la fois. De plus, ce projet nous a permis d'appréhender dans une certaine mesure les différentes phases d'élaboration d'un produit et malgré les difficultés rencontrées, nous avons réussi à aboutir à un résultat dont je suis très fière. La modélisation sur Solidworks m'a également beaucoup fait progresser en CAO.
J'avais initialement choisi ce projet par curiosité et pour son aspect "pratique", mais j'ai finalement développé, grâce à cette expérience, un réel goût pour la robotique. Hésitant encore entre les départements Mécanique et Energétique et Propulsion, ce projet va me permettre d'affiner mon projet professionnel.

Corentin PORIEL : Ce projet m'a donné l'opportunité de découvrir le monde de l'électronique. Coder la NodeMCU m'a également permis d'appréhender l'a programmation avec une approche différente de celle qui nous est enseignée avec le langage Pascal en cours. Hésitant encore en début de semestre entre les départements ASI et GM, ce projet a été l'occasion de confirmer ma préférence pour le premier.

Damien TOOMEY : Ce projet est une première approche en ce qui concerne la réalisation d'une application Android. MIT App Inventor est un moyen intéressant pour développer des applications mais il demeure nécessaire d'apprendre à utiliser les logiciels comme Android Studio pour devenir Développeur Android, ce qui correspond à mon projet professionnel.

Juliette VALLOT : Ce projet m'a permis de m'épanouir dans le domaine de la mécanique à travers la réalisation d'un robot. De plus, c'est un projet intéressant comme tous les membres du groupe étaient très motivés quand à la réalisation du projet. L'apprentissage de l'utilisation de l'imprimante 3D a également été très enrichissante à découvrir. Notre répartition du travail nous a montré à tous de bien savoir se servir des points forts de chacun au sein d'un groupe de projet, ce qui est très important en tant que futur ingénieur pour le travail en équipe.

Perspectives pour la poursuite de ce projet :

Il serait intéressant de réaliser une version IOS de l'application pour que le robot puisse être piloté à partir d'un Iphone, Ipad ou Ipod.

5. Bibliographie

Partie application Android :

App Inventor 2 - ESP8266 GPIO Control with Nodemcu LUA
https://www.youtube.com/watch?v=qWKcOnoyBzE

How to Build an Android App to Control Your WiFi Enabled Arduino
https://www.youtube.com/watch?v=ZH7ufemP8e0

How to make an android app to control a robot using MIT app inventor (bluetooth)
https://www.youtube.com/watch?v=li9YaUactCw

Arduino car wifi mit app inventor
http://randomnerdtutorials.com/esp8266-controlled-with-android-app-mit-app-inventor/

Garage wifi
https://community.particle.io/t/android-app-mit-inventor-2-spark-web-enabled-button/17170

Take Screenshot Extension
https://puravidaapps.com/screenshot.php

App inventor - how to - camera function
https://www.youtube.com/watch?v=bcqJPFXaAWA

App Inventor: Using the Clock as a Timer
https://www.youtube.com/watch?v=sdDcZfeCBXc

File Extension
https://puravidaapps.com/file.php

Tools Extension
https://puravidaapps.com/tools.php

Disable Button After 5 Uses
https://groups.google.com/forum/#!topic/appinventor/7BCvS0ee5BE

Partie application Arduino

Prise en main du module wifi esp8266 avec arduino
http://www.evola.cc/prise-en-main-du-module-wifi-esp8266-avec-arduino/

Tutoriel arduino d'OpenClassrooms
https://openclassrooms.com/courses/programmez-vos-premiers-montages-avec-arduino

Ip Webcam
https://play.google.com/store/apps/details?id=com.pas.webcam&hl=fr

Feuille de données de la carte TB6612FNG
https://www.sparkfun.com/datasheets/Robotics/TB6612FNG.pdf

6. Annexe- Code arduino commenté

```
#include <ESP8266WiFi.h>

const char* ssid = "G4_9284"; // nom du hotspot créé par le téléphone
const char* password = "Corentin";  //mot de passe du réseau
WiFiServer server(80); //port réseau utilisé pour la configuration d'un serveur HTTP


//Le numero des sorties ne correspondent pas avec ce qui est marqué sur la carte physiquement,
//on fixe donc des constantes facilitant l'utilisation des sorties.
#define D0 16
#define D1 5
#define D2 4
#define D3 0
#define D4 2
#define D6 12
#define D7 13
#define D8 15
#define D9 3
#define D10 1
#define D11 9
#define D12 10

//Les constantes suivantes permettent de faire le lien plus facilement entre les sorties de la
//NodeMCU V3 et celles de la carte TB6612FNG
const int PWM_A = D7; //Sortie reliée au moteur A
const int PWM_B = D12; //Sortie reliée au moteur B
const int AIN1 = D6;
const int AIN2 = D5;
const int BIN1 = D4;
const int BIN2 = D0;
bool ModePhoto = false; //permet d'ajuster la taille de la vidéo à l'écran lors du passage du mode
                        //photo au mode pilotage


void setup() {
  Serial.begin(115200); //vitesse de communication entre la carte et le moniteur série (sur le pc)
```

```
  delay(10);
  analogWriteFreq(20000);
  analogWriteRange(255);
  pinMode(PWM_A, OUTPUT); // Vitesse moteur D
  pinMode(PWM_B, OUTPUT); // Vitesse moteur G
  pinMode(AIN1, OUTPUT);  // Direction moteur D
  pinMode(AIN2, OUTPUT);  // Direction moteur G
  pinMode(BIN1, OUTPUT);  // 0 pour freinage
  pinMode(BIN2, OUTPUT);  // 0 pour freinage


  analogWrite(PWM_A, 0); // Arrêt complet moteur
  analogWrite(PWM_B, 0); // Arrêt complet moteur

  digitalWrite(AIN1, LOW);  // pour arrêt robot
  digitalWrite(AIN2, LOW);  // pour arrêt robot
  digitalWrite(BIN1, LOW);  // pour arrêt robot
  digitalWrite(BIN2, LOW);  // pour arrêt robot

   // Connexion au WiFi
  Serial.println(); //saut de ligne dans le moniteur série
  Serial.println(); //saut de ligne dans le moniteur série
  Serial.print("Connexion borne wifi : "); // indique sur le moniteur série que la connexion au
                                           //réseau
  Serial.println(ssid); //dont le nom est enregistré dans ssid s'apprête à avoir lieux

  WiFi.begin(ssid, password);

  while (WiFi.status() != WL_CONNECTED) { // tant que la carte n'est pas connectée on écrit
un
                                          //"." dans le moniteur série toutes les 1/2
secondes
    delay(500);
    Serial.print(".");
  }
  Serial.println(""); //saut de ligne
  Serial.println("Connexion WiFi etablie "); // on indique que la connexion est établie sur le
                                            //moniteur série une fois que c'est le cas

 // Demarre le serveur
  server.begin();
  Serial.println("Serveur en marche"); //on indique que le serveur est démarré dans le moniteur
```

```
                                                    //série
    Serial.print("Adresse IP pour se connecter au Robot : "); // indique l'adresse IP de la carte,
elle est peut également être récupéré grâce au téléphone sur lequel la NodeMCU V3 s'est
connectée
    Serial.print("http://");
    Serial.print(WiFi.localIP());
    Serial.println("/");
}

void loop() {

    // On vérifie si un client est connecté
    WiFiClient client = server.available();
    if (!client) { //on recommence le loop tant qu'aucun client n'est connecté
      return;
    }
    // attend des données du client
    Serial.println("Nouveau Client");
    while(!client.available()){
      delay(1);
    }

    // Lire la première ligne de la requête
    String request = client.readStringUntil('\r');
    Serial.println(request);
    client.flush();

    // On identifie la requête et appelle la fonction correspondante
    if (request.indexOf("/S") >0) {
        toutArreter();
    }
    else if (request.indexOf("/Av+") >0) {
        avancerRapidement();
    }
    else if (request.indexOf("/Av-") >0) {
        avancerLentement();
    }
    else if (request.indexOf("/D") >0) {
        tournerDroite();
    }
    else if (request.indexOf("/G") >0) {
        tournerGauche();
    }
```

```
    else if (request.indexOf("/T") >0) {
       faireDemiTour();
    }
    else if (request.indexOf("/R") >0) {
       marcheArriere();
    }
    else if (request.indexOf("/Photo") > 0) {
       ModePhoto = true;  // si l'utilisateur veut prendre une photo on fait passer le booleen
    //"ModePhoto" à "vrai"
    }


    //Le serveur envoie la page HTML au client
    client.println("HTTP/1.1 200 OK");
    client.println("Content-Type: text/html");
    client.println("");
    client.println("<!DOCTYPE HTML>");
    client.println("<html>");
    client.println("<head>");
    client.println("<meta name='apple-mobile-web-app-capable' content='yes' />");
    client.println("<meta name='apple-mobile-web-app-status-bar-style'
content='black-translucent'
                        />");
    client.println("<meta name='viewport' content='width=device-width, minimum-scale=0.1'/>");
    client.println("</head>");
    client.println("<body bgcolor = \"#e0e6ec\"; margin: 0;>");



    switch (ModePhoto){ //Dans le cas où...
      case 0:// ...le mode photo est desactivé
         client.println("<div style =  'text-align : center'>");//on centre ce qui suit
         client.println("<img src='http://192.168.43.1:8080/video' width='202.5' height='360'
                        /></div>");// on affiche en petit la vidéo
      break;
      case 1 : //...le mode photo est activé
         client.println("<div style =  'text-align : center'>"); // on centre ce qui suit

//on indique qu'il faut récupérer la vidéo à l'adresse d'IP WebCam('http://192.168.43.1:8080') et
//on l'affiche avec une largeur de 298 pixels sur 530 de longueur
         client.println("<img src='http://192.168.43.1:8080/video' width='298' height='530 '
                        /></div>");      break;
    }
```

```
    client.println("</html>");
    delay(10);
    Serial.println("Deconnexion Client");
    Serial.println("");

}

void avancerRapidement(){//Quand la fonction avancerRapidement est activée
     digitalWrite(AIN1, LOW);  //on active les moteurs
     digitalWrite(AIN2, HIGH);
     digitalWrite(BIN1, HIGH);
     digitalWrite(BIN2, LOW);
     analogWrite(PWM_A, 255);  // à leur puissance maximum
     analogWrite(PWM_B, 255);
  }

void avancerLentement() {//Quand la fonction avancerLentement est activée
     digitalWrite(AIN1, LOW); //on active les moteur
     digitalWrite(AIN2, HIGH);//dans le même sens qu'au dessus
     digitalWrite(BIN1, HIGH);
     digitalWrite(BIN2, LOW);
     analogWrite(PWM_A, 200);// mais à plus faible puissance
     analogWrite(PWM_B, 50);
}

void tournerGauche() {

     digitalWrite(AIN1, LOW);
     digitalWrite(AIN2, HIGH);
     digitalWrite(BIN1, LOW);
     digitalWrite(BIN2, HIGH);
     analogWrite(PWM_A, 255);
     analogWrite(PWM_B, 255);//on active le moteur droit
}

void tournerDroite() {

     digitalWrite(AIN1, HIGH);
     digitalWrite(AIN2, LOW);
     digitalWrite(BIN1, HIGH);
     digitalWrite(BIN2, LOW);
     analogWrite(PWM_A, 255);//on active les deux moteurs de manière
     analogWrite(PWM_B, 255);
```

```
 }

void faireDemiTour()  {
    digitalWrite(AIN1, HIGH);//on active les moteurs
    digitalWrite(AIN2, LOW);// dans des sens opposés
    digitalWrite(BIN1, HIGH);
    digitalWrite(BIN2, LOW);
    analogWrite(PWM_A, 255);// à puissance maximale
    analogWrite(PWM_B, 255);
    delay(400);//on attend quelques millisecondes
    analogWrite(PWM_A, 0); //on éteint tout
    analogWrite(PWM_A, 0);
    digitalWrite(AIN1, LOW);
    digitalWrite(AIN2, LOW);
    digitalWrite(BIN1, LOW);
    digitalWrite(BIN2, LOW);
}

void marcheArriere() {//comme avancerRapidement mais dans l'autre sens
     digitalWrite(AIN1, HIGH);
     digitalWrite(AIN2, LOW);
     digitalWrite(BIN1, LOW);
     digitalWrite(BIN2, HIGH);
     analogWrite(PWM_A, 255);
     analogWrite(PWM_B, 255);
}

void toutArreter() {//on met tout à 0
     digitalWrite(AIN1, LOW);
     digitalWrite(AIN2, LOW);
     digitalWrite(BIN1, LOW);
     digitalWrite(BIN2, LOW);
     ModePhoto = false; //Une fois la photo prise une requête /S est envoyée, pour prévenir qu'il
                        //faut remettre les petites dimensions d'affichage de la vidéo
                        //on passe donc le booléen "ModePhoto" à faux
}
```