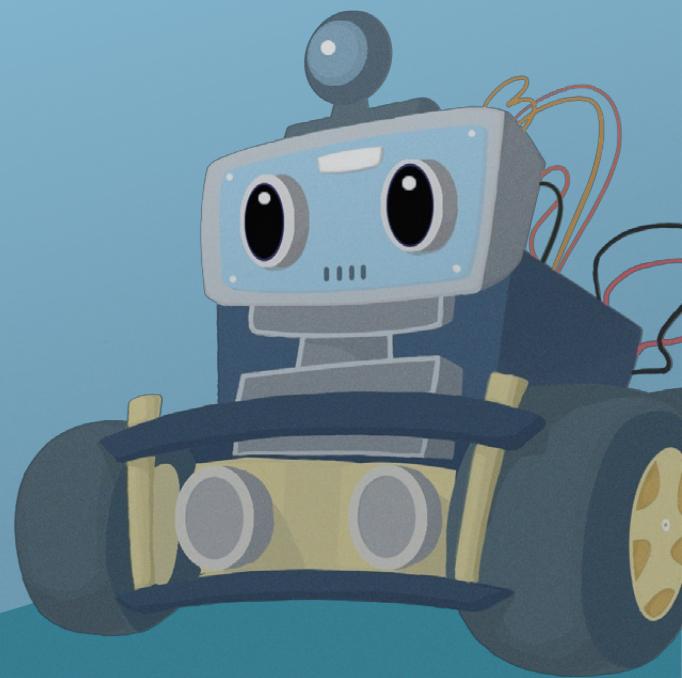
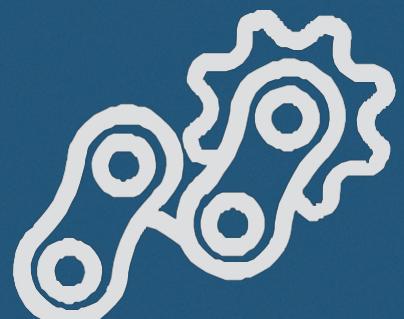


4

SMART ROBOT CAR V4.0 WITH CAMERA



Servo Control





Введение:

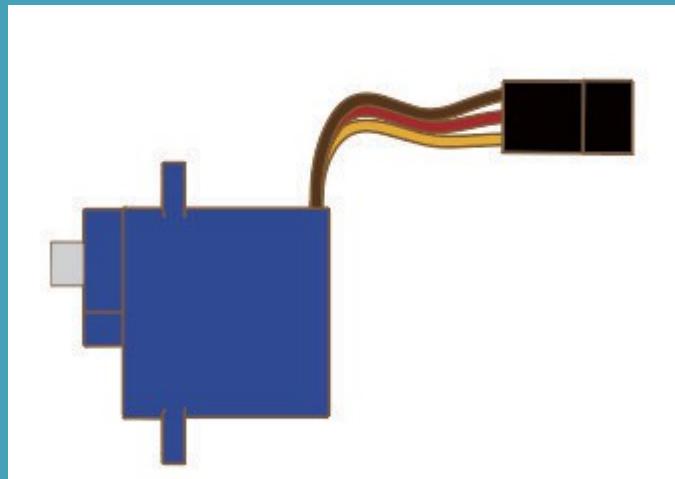
- + В этом уроке мы расскажем, как управлять сервоприводами Smart Robot Car.

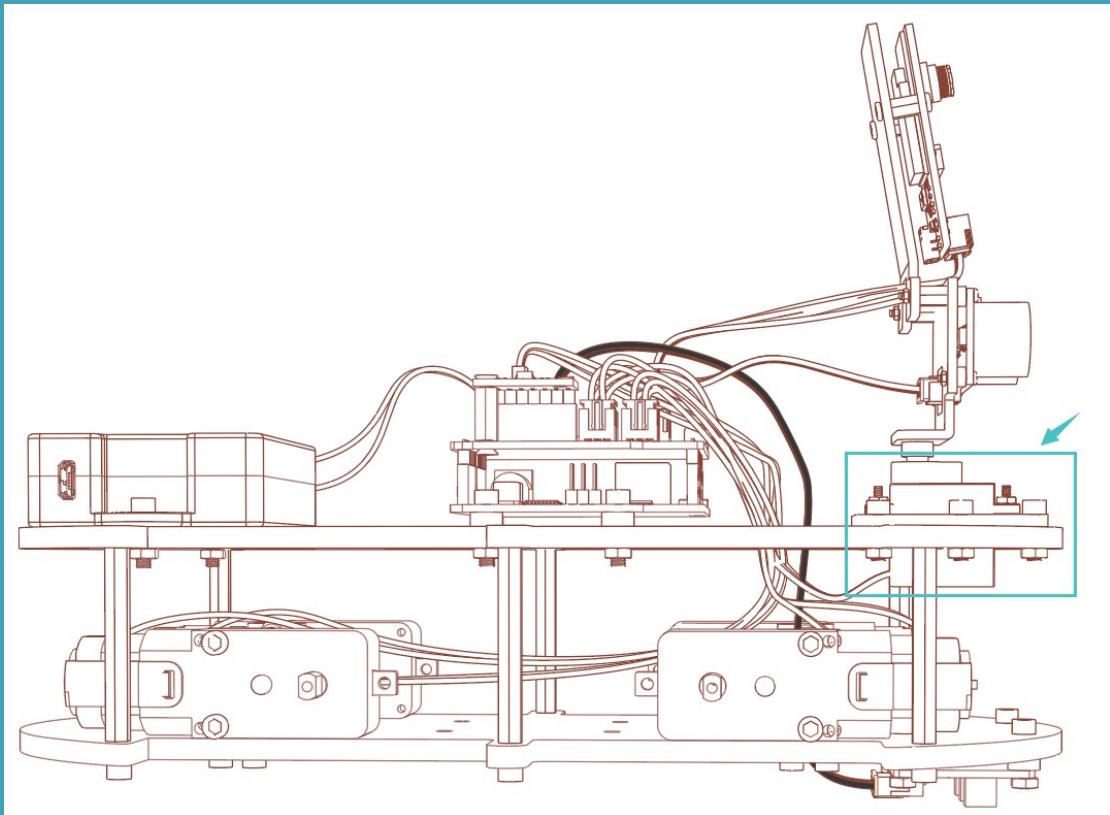


Подготовка:

- + Умный автомобиль-робот (с аккумулятором) Провод USB

+ Сервопривод - это тип двигателя, также известный как сервомотор. Он аналогичен шаговому двигателю. Разница между сервоприводом и шаговым двигателем заключается в том, что шаговый двигатель может задавать количество углов поворота, а сервопривод - положение поворота.

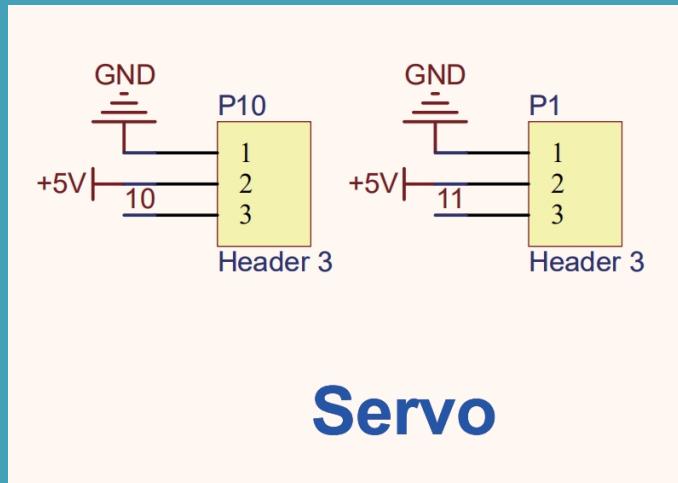




Для получения более подробной информации откройте файл последней папки: [Информация о сопутствующих микросхемах -> SmartRobot-Shield](#)



Как показано на рисунке, сервопривод может быть подключен к выводу D10 или D11 модуля UNO.



И управление сервоприводом также относительно простое. После добавления библиотеки `official` можно произвольно изменять угол управления.



Пожалуйста, откройте [Demo1](#) в текущей папке:

 Прежде всего, рассмотрим определение выводов и переменных УЗИП: (В данном примере мы подключили сервопривод к выводу D10).

C:/ // в DeviceDriverSet_xxx0.h

```
/*Servo*/ #include
<Servo.h>
класс DeviceDriverSet_Servo
{
    общественность:
        void DeviceDriverSet_Servo_Init(unsigned int Position_angle);
    #if _Test_DeviceDriverSet
        void DeviceDriverSet_Servo_Test(void);
    #endif
        void DeviceDriverSet_Servo_control(unsigned int Position_angle);
    private:
        #define PIN_Servo_z 10
};
```

 Сначала познакомимся с общими функциями

uint8_t attach(int pin)	присоединяет заданный пин к следующему свободному каналу, устанавливает режим pinMode, возвращает номер канала или 0 в случае неудачи
uint8_t attach(int pin, int min, int max)	как и выше, но также устанавливает минимальное и максимальное значения для записей
void write(int value);	если значение < 200, то оно рассматривается как угол, в противном случае - как ширина импульса в микросекундах
void detach()	Отсоедините сервопривод от его интерфейса, который можно продолжать использовать в качестве ШИМ-интерфейса

 Затем инициализируйте сервоприводы, подключенные к выводам D10 и D11, и установите положение сервоприводов на 90 градусов.

C:/ // в DeviceDriverSet_xxx0.cpp

```
Servo myservo; // создание объекта сервопривода  
для управления сервоприводом void  
DeviceDriverSet_Servo::DeviceDriverSet_Servo_Init  
(unsigned int Position_angle)  
{  
    myservo.attach(PIN_Servo_z, 500, 2400);  
    //500: 0 градусов 2400: 180 градусов  
    myservo.attach(PIN_Servo_z);  
    myservo.write(Position_angle);  
    //устанавливает положение сервопривода в соответствии с  
    //положением 90 (middle) delay(500);  
}
```

 А затем выбрать необходимый сервопривод в соответствии с углами поворота сервопривода.

C:/ // в DeviceDriverSet_xxx0.cpp

```
/*0.17sec/60degree(4.8v)*/  
void DeviceDriverSet_Servo::DeviceDriverSet_Servo_control  
(unsigned int Position_angle)  
{  
    myservo.attach(PIN_Servo_z);  
    myservo.write(Position_angle);  
    delay(450); myservo.detach();  
}
```

Наконец, загрузите программу. (При загрузке программы переключите кнопку "Upload-Save" на "Upload"). После успешной загрузки вы увидите, как сервопривод повернется на полкруга, а затем вернется в среднее положение.