

Challenges & Solutions In FTC Programming

אתגרים ופתרונות בתכנות FTC

מתן פורת  #1943, CSA



כמתכנתים אנו עשויים להיתקל במגוון רחב של בעיות .
בהרצאה נעבור על כלים שיעזרו להימנע מבעיות עתידיות בקוד:



- שימוש בהערות
- חלוקת הקוד לפעולות
- Debugging
- בעיות נפוצות ופתרונם



ישנן מספר דרכים לכתובת קוד לרובוט FTC:

1. Java

2. Blocks programming

3. App inventor



בהרצאה זו נתמקד ב**Java**



Comments

היתרונות של שימוש בהערות לאורך הקוד:

- מקל על העבודה עם אנשים נוספים
- מזרז את התהליך של בדיקת הקוד
- לא צריך להסתמך על הזכרון

REAL PROGRAMMERS
DON'T COMMENT THEIR CODE.

IF IT WAS HARD TO WRITE,
IT SHOULD BE HARD TO
UNDERSTAND.

Methods

חלוקה לפעולות



Methods

מתודה/פעולה היא תבנית שאנו כותבים מראש בידיעה
שנרצה לקרוא לה יותר מפעם אחת בקוד.
כך אנו חוסכים שורות קוד והופכים את הקוד למובן
יותר.

Methods

טיפוס ההחזרה

שם הפעולה

פרמטרים

```
int DigitsCount(int num) {  
    int count = 0;  
    while (num > 0)  
    {  
        count++;  
        num = num / 10;  
    }  
    return count;  
}
```

Methods

```
// Same exercise, but now with functions!  
int num1, num2, num3;  
cout << "Enter a number" << endl;  
cin >> num1;  
int count = 0;  
count += DigitsCount(num1);  
  
cout << "Enter a number" << endl;  
cin >> num2;  
count += DigitsCount(num2);  
  
cout << "Enter a number" << endl;  
cin >> num3;  
count += DigitsCount(num3);  
  
cout << "Digits count sum: " << count << endl;
```

```
// Reading 3 numbers and calc their digits count sum  
int num1, num2, num3;  
cout << "Enter a number" << endl;  
cin >> num1;  
int count = 0;  
while (num1 > 0)  
{  
    count++;  
    num1 = num1 / 10;  
}  
cout << "Enter a number" << endl;  
cin >> num2;  
while (num2 > 0)  
{  
    count++;  
    num2 = num2 / 10;  
}  
cout << "Enter a number" << endl;  
cin >> num3;  
while (num3 > 0)  
{  
    count++;  
    num3 = num3 / 10;  
}  
cout << "Digits count sum: " << count << endl;
```


Debugging

- Break Points
- Console
- Log





Break Points

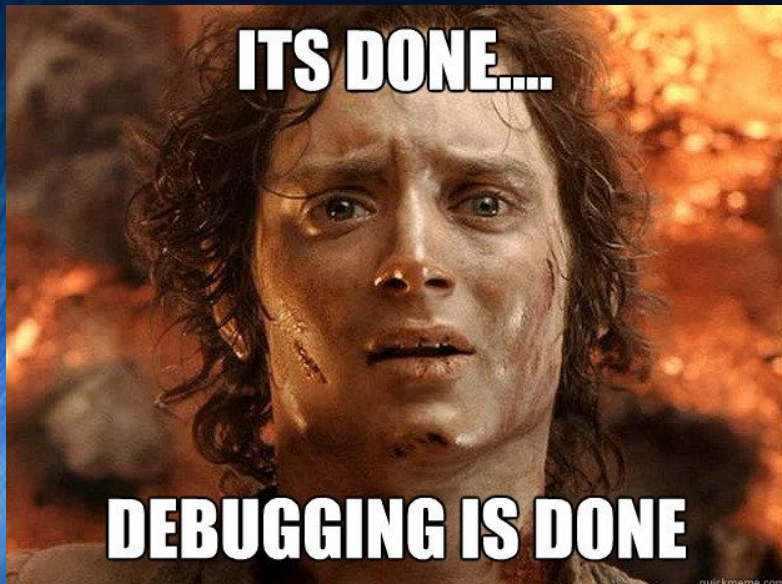
פונקציה שמאפשרת לנו כמתכנתים לעצור את הקוד בנק' ספציפית. מאפשר לנו להשיג מידע כמו:

- האם הריצה מגיעה לנקודה בה קבענו עצירה
- ערכים של משתנים בנקודה בה עצרנו
- בידוד של הגורם לקריסת הקוד



Console

הדפסה של מידע מהרובוט לעמדת הנהגים



- מידע בזמן אמת למשתמש
- בדיקת Flag
- בידוד של גורמים שונים בקוד

שאלות??



Common issues & solutions

Common issues & solutions

- **Bad** Configuration
- Case sensitive
- OpMode Register
- Hardware
- Connection



Configurations

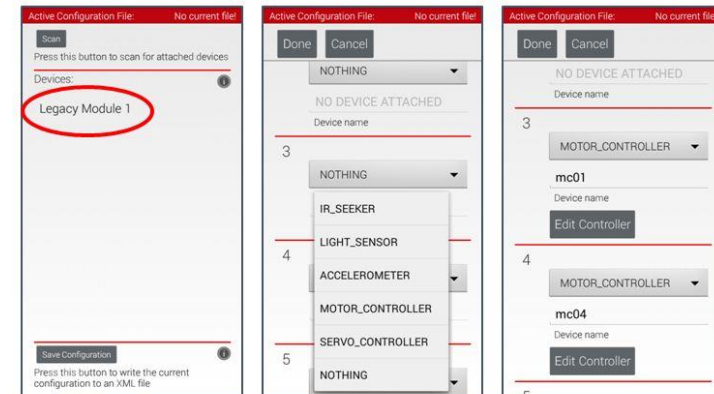
- Wrong port number
- Incompatibility
- Check the conf name

Programming the Legacy Module - Android Studio



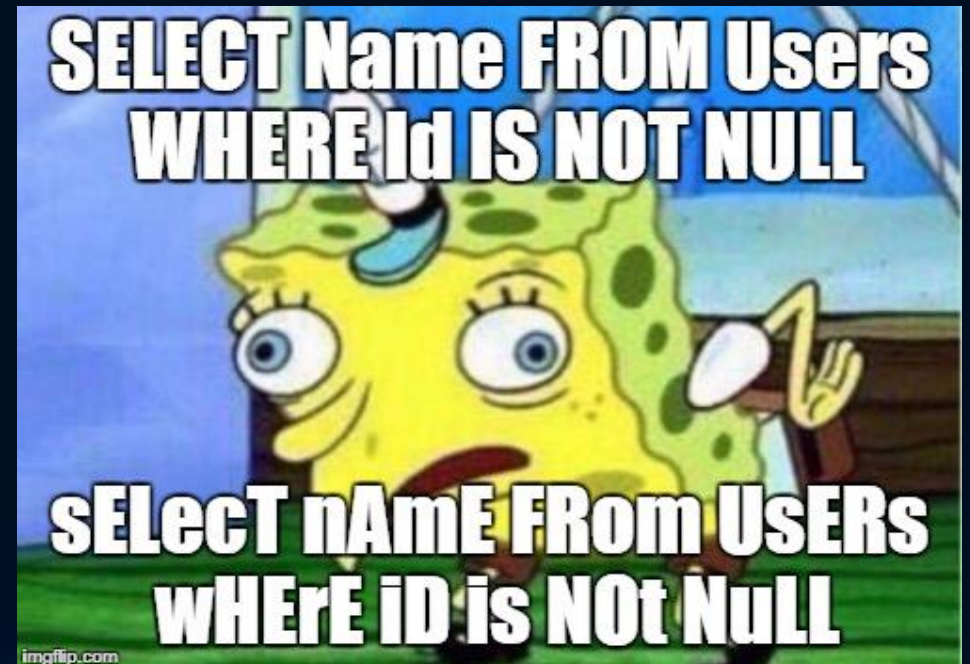
Create Hardware Configuration File

- Click on the Legacy Module and assign devices to connected ports



Case sensitive

- In your code
- Between the code & conf
- With the first package



OpMode Register

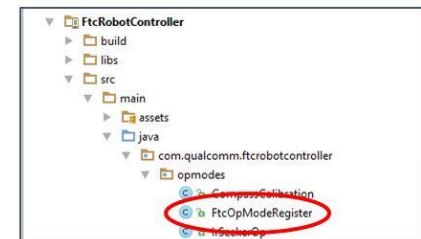
- OpModes were not registered
- The registration wasn't good

Programming the Legacy Module - Android Studio

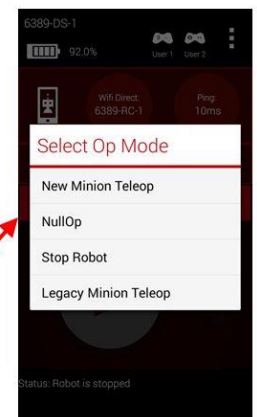


Creating a new Op Mode

- Add the new OpMode class to the list of registered OpModes to make it show up on the Driver Station App

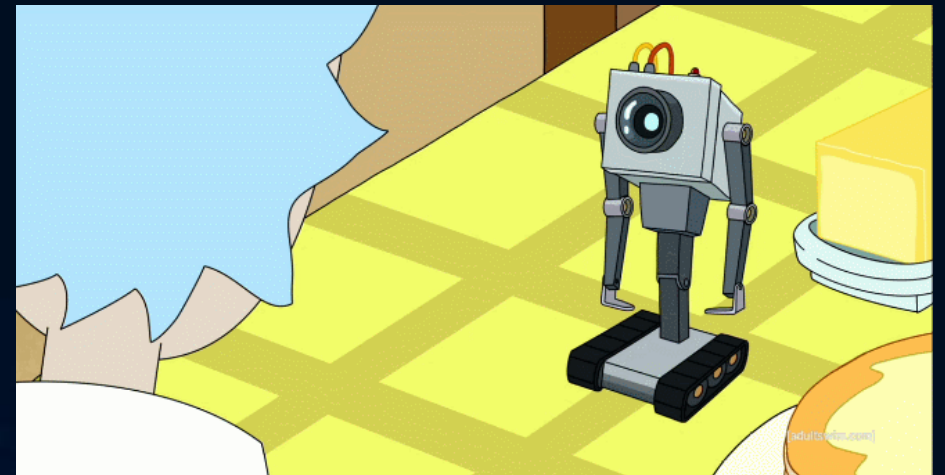


```
public void register(OpModeManager manager) {  
    manager.register("NullOp", NullOp.class);  
    manager.register("New Minion Teleop", NewMinionTeleop.class);  
    manager.register("Legacy Minion Teleop", LegacyMinionTeleop.class);  
}
```



Hardware issues

- Software **can't** fix hardware
- Hardware affects Software



No communication

- Bad ports definition
- Disturbances
- **Bad** phone positioning
(Faraday cage)





בבקשה תשאלו משהו (כל דבר)



Thanks a lot!



Email:

Matanp123@gmail.com