RoboBibb Programming Team

A quick introduction for prospective members

Who am I?

- Programming captain
- Creator of YodaScript
- Senior

- **Links:**
  - **GitHub**: github.com/dvtate
  - **Website**: dvtate.com
  - **Google+**: google.com/+tatetesta
  - **Telegram**: telegram.me/ridderhoff

Dustin Van Tate Testa
dvtate

Difficulty of Team

- Programming is one of the easiest teams with one of the steepest learning curves.
- We can generally finish the bulk of the robot code in only a few hours.
- Because programming is a use it or loose it skill, it is important that we work on projects year-round.
- Historically programming has been a small team however, more projects == more members
- I would prefer that new members have some knowledge of programming (any language).

What we make and skills required

- Websites and web pages (HTML+CSS+JS, GitHub)
- Arduino projects (Arduino/C++, circuitry)
- Chat-bots (node.js, bash shell, linux, regular expressions)
- FRC robot code (C/C++)
- [practice] programs and calculators (variety of languages, GitHub)
- Personal projects (¯_(ツ)_/¯)
- other useful skills: C/C++, GDB, JS, bash/batch, python, Java, Matlab, software licensing, GitHub, autohotkey, PHP, SQL, math, terminal productivity, linux, etc.

- I for one rarely work on more than one of these at a time. I generally cycle through areas of interest and its acceptable for you to do the same.

Communication with other teams

- Most important:
  - **Electrical**: port numbers, sensors, etc.
  - **Mechanical**: what our code has to do (what are we even controlling)
  - **Marketing**: website design/development, graphics, art, etc.
  - **Drive**: Controls and ideas
  - **Mentors**: help with math and stuff

- Less important:
  - **Scouting**: during competition we often join them in stands
  - **Safety**: programming is probably the safest team

- Working with everyone:
  - Telegram - Slack is our method of communication with mentors, but Telegram is more useful for us students, we made Steve the Telegram bot to manage the chats.
  - Website - We manage the website so if there needs to be a content change we are usually responsible
  - Friends - We're all friends here so let's be nice :)

How we code together – GitHub

- GiHub is a version control system
  - it's like Google Drive for programmers
- It's very useful for sharing open-source projects
- We have a team GitHub organizaton which contains our team projects
- in addition, github provides us with free hosting for our website (robobibb.github.io) via github-pages

Gaining skill

- We will have some meetings specifically focused on learning, however, you will need to do most of your learning at home.
- Join group chats (osmosis)
- wikipedia - binge read nerd stuff
- Do projects/challenges - I'll probably assign some
    - Making practice programs only goes so far
- use github - I want to see your progress
- Linux - more on this later
- See related slides for ideas on how to practice at home.

So you want to be a WebDev?

- **Best fit:** If you would otherwise get bored and want something pretty to show for your work. UI design and development is the most important skill to master here.
- **Job:** programming the website, making new pages, aesthetic
- **Practice:** make a personal website (using github-pages is fine). Make it cover a variety of things about you and have some functionality (ie- calculators, interactivity, etc.). It doesn't have to look pretty, functionality is more important. My website (dvtate.com) might give you some ideas (however I used PHP)
- **Difficulty:** easiest team, although you have to type large volumes of code, the concepts required to create it are straightforward, and compaired to the other teams the learning curve isn't so steep.
- **Best OS:** (1st) **Mac OS**, (2nd) linux, (3rd) windows. No single OS is unarguably better suited to what we're doing, however if/when we upgrade the website Windows will become less favorable.

So you want to program Arduinos?

- **Best Fit:** If you like retro game consoles and the cool hacks to get things working
- **Job:** Part of doing well at competition is looking cool. We use arduinos to control LED's and other things to do certain tasks.
- Unfortunately there isn't much work to do, and rarely the need for collaboration.
- **Difficulty:** intermediate. It's a fairly easy way to get into C/C++ and good practice for programming robots.
- **Practice:** make a night-light with an aduino and some RGB LED's. Make color/light patterns.
- **Best OS:** all OS's are **equal**ly suited as Arduino IDE is cross-platform

So you want to make chatbots?

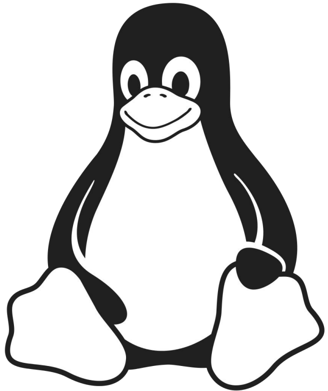
- **Best Fit**: If you ever wanted to have a robot friend to talk to. If you are proud of making something work in an innovative way.
- **Job**: managing a Linux server and a Telegram bot.
- There are so many things we can do with this, it can have unlimited integrations and can grow into something genuinely useful and fun to interact with.
- **Difficulty**: Advanced, You need experience with a Linux terminal, Shell scripting, regular expressions, and asynchronious node.js. On the bright side, much of the infastructure, (shell scripts and linux server), has already been established and are relatively low maintainence.
- **Best OS: Linux** is really the only reasonable option here. Windows would be the worst as it doesn't natively support many essentials.
- **Practice**: Install Linux on your primary machine. Use it as your daily driver. Make your own telegram bot using the API which performs some basic function.

So you want to program robots?

- **Best Fit**: you want a serious challenge and be able to rapidly produce quality code. (ie- competitions)
- **Job**: You will be writing the code that our robot runs at competition.
- **Practice**: since this group will be our creme of the crop, we will be practicing C/C++ frequently at meetings. If get any projects which involve Arduinos this would also be good practice.
- **Difficulty**: Challenging, although the code we have been running for the past few years isn't very difficult, this year I want to make some more advanced code. In addition, most of the language features we use are in the last few chapters of most C++ books. And with C/C++ you really want a solid base.
- **Best OS**: although most of the software works most of the time on Linux/OSX, it's designed for **Windows** and works best for it so we will have to use it.

Linux

- Linux is a programmers' paradise.
- It's lightweight and open-source, making hacking easier
- It's what I use and what the robot is running.
- If you have an old PC I can help you install linux on it. (free)
- Its definitely not for everyone but is a useful tool.



Questions? Comments? Concerns?

- If there's anything that seemed interesting to you and we still have time I'd be happy to go over that now.