Bot Specification Sheet

A guide to 'pp-cli'

dkantereivin, Doomer

April 4, 2021

Contents

Chapter 1

Introduction

This is the Bot Specification - here every feature and their usage will be documented! In the first chapter the core structure of the bot is explained in detail and some decisions concerning the project organization are discussed. Apart from general design decisions, Chapter 1 will also include a quick overview over the project goals and milestones for the near future.

Chapter 2

Core Structure

'pp-cli' consists out of 3 components, which are crucial for it to work! Events Handlers, Commands and Services are the three backbones of this project.
Event Handlers take Events from the Discord API and work with them - for example, fetch them to detect and eventually trigger commands or use them to classify data for a future machine learning subproject!
Commands are the part of the project that is triggered by certain events and usually processes and outputs a user message or accesses and possibly alters the database (soon to be implemented)!
Services - the final piece of that triplet is responsible for providing universal, reusable code snippets and are categorized into specific areas of application, some of them are needed for the bot to start up, while others take care of the functions during the bots runtime!

Chapter 3

Event Handling

Let's start with the first concept - the Event Handlers. The Event Handlers are quite simply organized structure-wise. The interface used as a blueprint for the event handlers contains a single field - `EVENT_NAME` and a single `onEvent` function, which automatically adapts to the arguments required for the specified Event Name such as seen in Listing 3.1!

```
interface IEventHandler<EventName extends keyof ClientEvents>
{
    readonly EVENT_NAME: EventName;
    onEvent(...args: ClientEvents[EventName]): void;
}

```

Listing 3.1: Event Handler Interface

At the moment we implemented 3 such event handlers. Ready Handler, for which the event gets fired at the bot start up. Message Handler, for which the event fires when the bot receives a message and a Message Reaction Add Handler, which handles incoming reactions for cached messages! All those play an important part for the application as a whole, and enable certain feature sets.
The Ready Handler prepares the Bot for action - this will be used in the near future to initialize the caches. The current implementation is as easy as the following:

```
1 public readonly EVENT_NAME = 'ready';
2 public onEvent() {
3     Logger.info('Connected.');
4 }
5
```

Listing 3.2: Ready Handler

The Message Handler is responsible for testing whether a command triggers given the message supplied by the event. The Reaction Handler tests for certain tags and if those apply it flags the message and adds it to the database (to be implemented)!

Chapter 4

Commands & Triggers

Chapter 5

Services