Checkmate Documentation

Release 0.1

Andreas Dewes

April 13, 2014

Contents

Checkmate is Python tool for code quality management. It gives you a global overview of the code quality in your project and provides you with actionable insights and advice.

Most current command-line tools for code quality management overburden the user with too much information, are hard to set up and adapt to the users needs and do not provide enough actionable insight.

Checkmate is an attempt to change this:

- It doesn't overwhelm your with tons of irrelevant information.
- It provides clear, actionable advice and a global view of your code quality.
- It wraps several code checking tools, providing a consolidated view on their data.

CHAPTER 1

Key Features

- Global statistics and issue lists for your project.
- Actionable, prioritized advice on how to improve the code.
- Built-in support of popular code checkers (**pep8**, **pylint** and **pyflakes**)
- Easily extendable and adaptable to your needs.
- Tracking of metrics and issues over time.
- Automatic versioning of your source files.
- Built-in support for git repositories.

CHAPTER 2

Installation

The easiest way to install checkmate is via **pip** or **easy_install**:

```
pip install 7s-checkmate
```

or

```
easy_install 7s-checkmate
```

Alternatively, you can just download the source from Github and install it manually by running (in the project directory):

```
git clone git@github.com:adewes/checkmate.git
cd checkmate
python setup.py build
sudo python setup.py install
```

CHAPTER 3

Quickstart

After installation, you should have a new **checkmate** command available in your shell. The command line interface (CLI) of checkmate is similar to that of **git**, so if you're familiar with the latter you should feel at home pretty fast.

3.1 Creating a new project

To create a new checkmate project, just go to your project folder and type:

```
checkmate init
```

If run inside a git repository, this will create a new checkmate project in the root folder of the git project. Otherwise it will create a project in the current directory. Like *git*, *checkmate* keeps all its files in a *.checkmate* folder.

3.2 Analyzing your code

To analyze your projects and produce metrics and issue lists, simply type:

```
checkmate analyze
```

This will generate of relevant files in your project, analyze them and generate a summary of the results. You can use on of the following commands to work with the results of the analysis:

```
#Shows a summary of statistics and issues for your project
checkmate summary
#Show a list of issues for your project
checkmate issues
#Show a summary of the statistics & metrics of your project
checkmate stats
```

There are some other commands you might find useful, for more information about the available commands or a specific command, just type

```
#Provides an overview of all commands and options
checkmate help
#Provides information about a specific command
checkmate help [command name]
```

3.2.1 API

Checkmate is designed in a way that makes it easy to use it both as a command line tool, or as a library.

Before you dive into the details of the API, be sure to read the section on the *core concepts*, which will give you a high-level overview of the library.

Core Concepts

Checkmate uses three core abstractions:

- **Project** : A project is everything that structure that contains a *.checkmate* directory.
- **Snapshot** [A snapshot represents the state of an entire project or a subset of it at a given point,] which can e.g. be a given commit or the local state at a given point in time.
- **File Revision** : A file revision represents the state of a given file in the project at a given point.

The Project

In checkmate, you create a new project by running *checkmate init*. This will create a *.checkmate* folder either in your current working directory, or, if you call it inside a git repository, in the directory that contains the *.git* folder. You project consists of all files contained in the directory that contains the *.checkmate* folder (including all subdirectories).

Snapshots

A snapshot represents the state of your entire project or a subset of the project at a given point. This can be e.g. a certain git commit or just the local state of the project directory at a given time. Snapshots are generated through the *checkmate analyze* command.

File Revisions

A file revision describe the state of a file at a given point. It contains both statistics as well as a list of issues for the given file. To create a snapshot, several file revisions are bundled together.

Generating a New Snapshot

When you run *checkmate analyze*, the following things happen behind the scenes:

- checkmate generates a list of all the files matching your analysis request
- it checks which of these file revisions have not yet been analyzed
- it analyzes the missing file revisions and stores the results in its database
- it creates a snapshot from the file revisions

Like this, checkmate makes sure to never analyze the same file twice, which greatly improves the performance of the tool when running it against a large codebase.