Automated Source Code Quality Checking

Type: Type 2
Christopher Freas and Jillian Jones
CSC 8350: Advanced Software Engineering
Department of Computer Science, Georgia State University
Spring, 2018

Background

As a software project grows in size, bugs and errors will inevitably creep into
the code. Additionally, the project may require an expanding team which can
lead to different styles of authoring code. Adding additional developers can
compound the amount of bugs introduced into the code since a developer may
struggle to understand the code that other developers have written and could
introduce new problems based on this lack of understanding.

In response to this, tools for many popular programming languages have
been introduced that can analyze code for potential bugs, formatting mistakes,
and can enforce code style standards. In this project we examine such a tool
for the popular Python programming language.

Overview of the hands-on project

The goal of the project is to understand how automatic code quality checking can
help improve software deployment by taking an in-depth look at the Python-
based Pylint code analysis tool. Pylint performs source code analysis, bug
checking, and code quality rating on Python code [2]. The output of the tool
shows the user the line numbers of any errors and style violations that have
occurred in the source code files and recommends the appropriate corrections.
The style checks are based on the well-known Python Enhancement Proposal
number 8 (PEP-8) [3], authored by the creator of Python, Guido van Rossum.
Importantly, the Pylint tool can be integrated into the Jenkins CI/CD server
[1]. By adding code quality checking to the CI/CD pipeline, certain standards
can be enforced, preventing code that does not meet these standards from being
added to any deployment packages. This enforcement of standards is what we
will explore and analyze in our hands-on project.
References

