ABSTRACT

The education of young and novice programmers is an ever-evolving field due to the prominence of new programming languages in the information industry. Current introductory programming environments are dominated by web-based tools such as MIT's Scratch service (Scratch Foundation, 2021). The goal of the Scratch platform is to offer engaging learning content targeted at younger users and children through block based programming in order to introduce the basics of programming logic and hopefully encourage the next generation of programmers. However, there is a gap in this educational field from the step from Scratch-like environments into the industry standard tools that are taught and used in Computer Science courses at third level education today. These introductory platforms are also only used to form a base knowledge of programming logic in the minds of new programmers where there is potential for introducing other industry standard practices such as Test Driven Development.

Pytch, which forms its name from a combination of 'Python' and 'Scratch', is a web-based programming environment centred around the education of the Python language through editing and compiling python code in browser, but also offers Scratch-like block component descriptions of its functions. This allows the platform to be a steppingstone platform for novice programmers with experience with Scratch to better adjust themselves to actual code editing in Python, rather than using the Scratch block based framework. The currently under development Pytch platform already has a series of pre-built projects that can be run as fully completed projects in the Pytch environment or they can be experienced through Pytch's tutorial environment which offers a step by step guide on the chronological building of the project.

This project analysed the current tutorial structure of Pytch by studying its design from both a frontend website and also backend application perspective through what documentation material was available and also from investigating the file and system structure manually. The goal of this project is to provide a feasible evolution of the Pytch tutorial system to include tutorial chapters that include practices of Test Driven Development. This is achieved by gaining an understanding of Pytch's tutorial design and structure so that new chapters and new features could be integrated into an existing Pytch tutorial.

This project implemented and evaluated the use of Python assertion tests within the Pytch environment to ensure that they communicate correctly with the Pytch web terminal. A visual proof of concept was then created by modifying the html of a currently existing tutorial chapter to evaluate the best practice for delivering the information on Test Driven Development. Finally, a functional proof of concept was created that offered an existing Pytch tutorial with chapters on Test Driven Development integrated at certain parts of the tutorial. The tests were successfully implemented along with their corresponding tutorial chapters and an outline of what was learned about forming the optimal tests for inclusion in educational work intended for novice programmers was also included.

The results of this project show that the inclusion of Test Driven Development is possible within the existing Pytch educational environment and also shows on a wider scope that it is feasible for developers of educational projects to include an introduction to Test Driven Development in their work by forming tests to aid in not only the education of the language they are working with, but also allows them to design these tests to be used for checks on the student's code that are closely related to the individual project rather than simply using built in syntax error checking that would come with a standard compiler.