Tap Here

A Diff-powered Programming Tutorial Engine

Michał Moskal Nikolai Tillmann Jonathan de Halleux Sebastian Burckhardt Thomas Ball
Judith Bishop

Microsoft Research
{micmo,nikolait,jhalleux,sburckha,tball,jbishop}@microsoft.com

Abstract

We discuss an interactive tutorial engine implemented in TouchDevelop [2, 3], a browser-hosted touch-based development environment for creating mobile+cloud applications. The tutorials are written as programs with directions embedded in comments, á la literate programming, and then automatically turned into direct and adaptive guidance to the user. The engine was deployed during the Hour of Code [1] event in December 2013 and used by more than 250,000 people, completing over 3,000,000 steps.

1. Background

Touch-based computing devices outsell traditional desktops and laptops by a wide margin; smart phones and tablets are often the first and sometimes the only computing device young people have. TouchDevelop [2, 3] is an integrated development environment (IDE) that allows programming of such devices directly, without a physical keyboard. The IDE runs in all major mobile and desktop web browsers (on Android, iOS, Windows Phone, Windows, OS X, and Linux). TouchDevelop includes a custom, statically-typed programming language, an editor, compiler, libraries, and a social forum of users, followers, comments and publishing of TouchDevelop scripts (programs).

The original goal of TouchDevelop was to bring **general-purpose** programming capabilities to **anyone** with a mobile touch-based computing device. The stress on "general-purpose" is reflected in the richness of the language, a large API surface (about 1500 built-in high-level APIs), and advanced features like a stepping debugger and crowd-sourced code coverage. The constraints of smartphone screen size and a touch-first interface led to a streamlined and focused IDE design, which in turn lends itself to education.

2. Interactive Tutorials

Tutorials build on top of TouchDevelop support for user-contributed documentation. Documents are simply TouchDevelop scripts where comments use markdown notation (a simple and commonly used markup language), extended with features specific to TouchDevelop (e.g., references to declarations, scripts, APIs). The comments render as the text of a documentation topic, and the remaining statements render as code snippets, á la literate programming.

Documents can be marked as interactive tutorials, which consist of *steps*; each step defines a target script, which we wish the user to create, and documentation describing the purpose of the step. A generic tutorial engine guides the user through the steps.

Typically, a step adds just a few lines of code over the previous step. The tutorial engine displays the documentation text for the step as a modal dialog and then guides the user to modify their script to match the target script. The modification may involve adding or deleting tokens and declarations, and navigating between different screens in the IDE.

An on-screen hint in the IDE points at the next button to tap to bring the working script closer to the target script. After the user has successfully completed a few steps, the hints are delayed and the desired form of the current line of code is displayed. The tutorial engine itself is stateless; it continuously computes the difference of the abstract syntax trees (AST) of the working and the target script, and observers the state of the IDE in order to determine the next hint. This flexibility allows the user to wander "off task" and explore different parts of the IDE or modify the working script in any way they wish—the engine will adapt and provide the necessary hints to bring the user back "on task" when they are done exploring.

Typically, programming language and API tutorials instruct the user to try out a few lines of code, modify something, try it again and so forth. Our tutorials codify this process in steps, with instructions interleaved with modifications. This has a number of advantages:

- there is no need to switch screens between the tutorial and the IDE, which is particularly useful on smart phones where switching screens is tedious and distracting;
- the tutorials teach the use of the editor user interface;
- the tutorials can be followed by users with no knowledge of programming, who nevertheless get a working app in the end; yet the step description provides in-depth information for users willing to learn;
- the matching between working script and target script can be arbitrarily fuzzy to allow user exploration.

We have collected data to support some of the points above. In particular, we track user progress and drop-out rates for every tutorial, total time spent in each step, time spent reading the description (most users don't read them; some do), editing, and running the program.

The experience with the tutorial engine shows how a simple programming language technique (AST diff) can be deployed to the massive benefit of the users inside the TouchDevelop platform. This suggests more advanced program analysis techniques could be even more useful.

We invite the reader to explore the tutorials developed for the 2013 Hour of Code event at `http://www.touchdevelop.com/hoc` to get a first-hand experience of the system. Many other tutorials are available from the TouchDevelop hub.

References

[1] Hour of Code Website. URL `http://csedweek.org/`.

[2] TouchDevelop Website. URL `https://www.touchdevelop.com/`.

[3] N. Tillmann, M. Moskal, J. de Halleux, and M. Fahndrich. TouchDevelop: programming cloud-connected mobile devices via touchscreen. In *Proceedings of the 10th SIGPLAN symposium on New ideas, new paradigms, and reflections on programming and software*, ONWARD '11, pages 49–60, 2011.