TUIOFX-Toolkit Support for the Development of JavaFX Applications for Interactive Tabletops

Mirko Fetter, David Bimamisa

TUIOFX—Toolkit Support for the Development of JavaFX Applications for Interactive Tabletops

Mirko Fetter, David Bimamisa

Human-Computer Interaction Group, University of Bamberg, Germany
(<firstname>.<lastname>(at)uni-bamberg.de)

Abstract. TUIOFX is a novel toolkit for developing multi-touch, multi-user applications for interactive tabletops and surfaces. By seamlessly integrating with JavaFX, TUIOFX provides a low entry barrier for developing state-of-the-art applications with multi-user, multi-touch capabilities and allows the cross-platform deployment on various interactive tabletop and surface hardware.

Keywords: Multi-touch, Interactive Tabletop, Toolkit, SDK, JavaFX.

1 Introduction

While the hardware for interactive tabletops and surfaces is getting more and more affordable, these technologies did not yet find a broad adoption in productive settings, beyond being an attention getter in exhibitions, showrooms and hotel lobbies. The absence of off-the-shelf applications for such systems can be considered as one barrier for technology adoption. An explanation for this absence is that the development for interactive surfaces often requires highly specialised knowledge and tools, and applications need to be tailored to specific hardware. With TUIOFX, a new toolkit that leverages on existing knowledge and standards aims to speed up the development of multi-user, multi-touch applications for a variety of hardware setups.

Figure 1. Three demo apps realised with TUIOFX in a few lines of code (LOC): (1) a multi-user video viewer (~100 LOC), (2) a drawing app (~450 LOC), and (3) a 3D model viewer (~400 LOC).

The TUIOFX toolkit was conceived to enable developers to easily develop appealing cross-platform applications (cf. Figure 1). To achieve this, TUIOFX combines the strengths of multi-user, multi-touch toolkits by supporting the TUIO protocol [3] with an extensive and fully stylable library of standard widgets (i.e. JavaFX UI controls). By building on JavaFX—and hence the Java programming

language—TUIFOX builds on Java's cross-platform support and the wide-spread knowledge of one of the most widely used programming languages.

In the following we highlight the requirements that guided the design of TUIOFX. We look at structure and implementation of TUIOFX and how programmers can utilise its' possibilities. We conclude with a discussion in respect to the related work.

2 Requirements and Design Decisions

Besides the overall goal of platform independence, the design of the TUIOFX was lead by *three* main requirements:

The first requirement was to *build on existing technology* to minimise the learning effort and maximise the available support. From our own experience with various toolkits, we often learned that those can be niche solutions, with sparse documentation and support, and sometimes even a limited lifespan. By building up on JavaFX we leverage on a huge, active community and a well-documented base technology. Through the minimal invasive design of TUIOFX a great number of tools—like the JavaFX Scene Builder for visual editing (cf. Figure 2)—seamlessly can be used for developing multi-user, multi-touch applications. By smoothly implementing everything beneath the surface, the documentation and support requirements for TUIOFX could be kept minimal. From our experience so far the transition effort— even for inexperienced JavaFX programmers—is negligible.

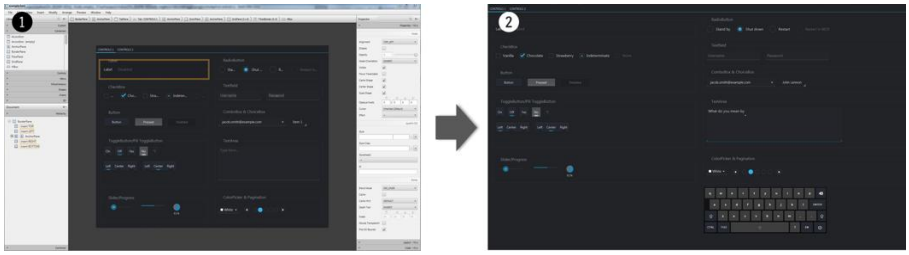

Figure 2. A TUIOFX GUI visually edited in Scene Builder (1) and the resulting application (2) with a touch enabled widgets including a soft-keyboard.

The second requirement was to provide an *extensive widget toolkit out of the box*. A great drawback of many existing toolkits is the limited amount of widgets that are provided, as the toolkit designers often need to implement each widget from scratch. By building on JavaFX, we are able to leverage from the many existing widgets from simple buttons to complex UI controls like charts, HTML editor, color picker, etc. However, the realisation required a major effort to adapt and optimise all existing widgets for touch on large interactive surfaces—including adding the possibility for text-entry through soft-keyboards, etc. By utilising JavaFX's concepts of skins and behaviours, this could be achieved while keeping possibilities for reusability, extensibility and stylability of the widgets intact. For a programmer, using a TUIOFX widget is no different than using the standard JavaFX API—all adaptations are made transparent for the programmer by loading the TUIOFX styles in the background.

The last requirement was the *support of multi-user interaction*. While JavaFX basically supports multi-touch, the design of JavaFX departs from a perspective of a single user on a hand-held device. To support multiple users on large interactive surfaces, TUIOFX adds various modifications to the standard behaviour of widgets. This includes solutions for: handling simultaneous user actions; reducing multi-user conflicts resulting from JavaFX single-focus model; free orientation of widgets; etc.

In the following more details on the implementation of TUIOFX are provided, as well as a few insights on how the toolkit is utilised by developers.

3 The TUIOFX Toolkit

The TUIOFX Toolkit is composed out of two parts, the TUIOFX–Core and the TUIOFX–WidgetToolkit. To migrate any existing JavaFX project into a multi-user, multi-touch TUIOFX project, both libraries can be easily added as JARs to a project.

TUIOFX–Core provides an abstraction from the raw TUIO messages and transforms them into standard JavaFX Touch- and Gesture-Events. TUIOFX core therefore uses JavaFX's multi-touch event handling infrastructure and its predefined `TouchEvent` and `GestureEvent` types in order to send events about recognised touches and gestures to the associated target `Node` (i.e. the UI element). TUIOFX's platform-independent touch and gesture recognizers are able to process TUIO messages into the corresponding event types for the JavaFX event platform, which are `TouchEvent`, `RotateEvent`, `ScrollEvent`, `SwipeEvent` and `ZoomEvent`. To achieve this TUIOFX analyses the global stream of TUIO touch events grouped per `Node` and sends events about recognised touches and gestures to the specific target `Node`s. Additional support for tangibles is ensured via a custom listener-interface, that simply forwards TUIO tangible object messages.

The TUIOFX–WidgetToolkit provides multi-user (e.g., text-fields that allow simultaneous text-entry through multiple on-screen soft keyboards) and multi-touch (e.g., UI controls that are adjusted in size for touch on big screens) optimised widgets. All changes are realised though extensive skinning. JavaFX allows overwriting the default look-and-feel of a `Control` by implementing custom `Skin` and `Behaviour` classes. This allows for complex adaptations of a UI element—like adding our own soft-keyboard to the `TextField` component—without any changes for developers on how to use this component in their code. Accordingly, we were able to keep the API of TUIOFX very lightweight, allowing the toolkit to be minimally invasive to existing code. The `TuioFX` class with only seven methods is the main point of interaction with the API. Two further classes (`Configuration` and `Configuration.Builder`) allow tailoring parameters of the gesture recogniser to specific hardware with several presets. And finally the class `TuioTangible` and the `TangibleListener` provide possibilities to deal with Tangibles.

By adding the two lines `TuioFX tuioFX = new TuioFX(stage)` and `tuioFX.start()` to the code, any existing JavaFX application immediately is able to react to TUIO input. With the line `tuioFX.enableMTWidgets(true)` widgets will loose their default look-and-feel and gain capabilities like soft-keyboard input, etc. This already provides the developer with 99% of TUIOFX's functionality

for developing multi-user, multi-touch applications. Beyond that, only understandings of a few custom properties are needed, for further fine-tuning the TUIOFX behaviour.

4 Related Work and Discussion

While a variety of multi-user, multi-touch toolkits exists, (e.g., libTisch [1], DiamondSpin [5], PyMT [2])—some even Java-based (e.g., MT4j [4])—most of them only provide a few custom widgets (e.g., button, text field). TUIOFX leverages on the several dozen, standard UI controls of JavaFX (incl. charts, tables, HTML5 editor, etc.), which are all fully skinnable with CSS. This further allows using tools like Scene Builder for visually developing multi-touch, multi-user applications and removes the need to learn highly specialised APIs. So far, the only comparable extent of functionality can be found in the Microsoft Surface 2.0 SDK (in combination with Expression Blend). However, the resulting applications are lagging a cross-platform support and are only running under Windows and on PixelSense or Surface hardware. TUIOFX applications can be deployed on a variety of hardware and software combinations as long as they support the TUIO protocol and Java 8.

We provided insights in the TUIOFX toolkit, a novel toolkit for developing multi-touch, multi-user applications for interactive tabletops and surfaces with JavaFX. We were able to collect first feedback by handing out TUIOFX to a small number of developers, which were quickly able to develop their own small applications. In a next step we plan to publicly release a first version of the TUIOFX.

Acknowledgments

We thank the members of the Cooperative Media Lab, especially Tom Gross.

References

1. Echtler, F. and Klinker, G. A Multitouch Software Architecture. In *NordiCHI 2008* (Lund, Sweden). ACM Press, NY, USA, 2008. pp. 463-466.
2. Hansen, T.E., Hourcade, J.P., Virbel, M., Patali, S. and Serra, T. PyMT: A post-WIMP Multi-touch User Interface Toolkit. In *ITS 2009* (Nov. 23-25, Banff, Alberta, Canada). ACM Press, NY, USA, 2009. pp. 17-24.
3. Kaltenbrunner, M., Bovermann, T., Bencina, R. and Costanza, E. TUIO - A Protocol for Table-Top Tangible User Interfaces. In *GW 2005* (May 18-20, Ile de Berder, France). 2005.
4. Laufs, U., Ruff, C. and Zibuschka, J. MT4j – A Cross-platform Multi-touch Development Framework. CoRR 2010.
5. Shen, C., Vernier, F.D., Forlines, C. and Ringel, M. DiamondSpin: An Extensible Toolkit for Around-the-table Interaction. In *CHI 2004* (Apr. 24-29, Vienna, Austria). ACM Press, NY, USA, 2004. pp. 167-174.