A web-based toolkit for remote direct manipulation interaction with public displays via smartphones

Jorge C. S. Cardoso
[1]CITAR/School of Arts
Portuguese Catholic University
Porto, Portugal
jorgecardoso@ieee.org

Maria Barreira[1,2]
[2]Faculty of Engineering
University of Porto
Porto, Portugal
mjoao.b3@gmail.com

ABSTRACT

Public displays are becoming increasingly interactive but we still need better programming tools that take the burden of adding interaction features to display applications. In this paper, we present a toolkit for web-based, remote, direct manipulation interaction with public display applications. Our toolkit provides high-level controls such as joystick, text input, and multi-touch cursor events, allowing programmers to focus on the interaction experience of the application.

Categories and Subject Descriptors

D.3.3 [**Software Engineering**]: Design Tools and Techniques – Software libraries, Modules and interfaces

General Terms

Human Factors

Keywords

Interactive public displays; toolkit; remote direct-manipulation interaction; smartphone

1. INTRODUCTION

Mobile devices have been widely used to interact with public displays. SMS/MMS messages [6], Bluetooth naming [4], custom applications [8] have all been used to provide some level of control over public displays. However, the concern is usually to create one display system that demonstrates the capabilities of the interaction technique. If interactive public display applications are to become widespread, we need to provide toolkits that developers can easily use to add interactive capabilities to their applications. In this work, we propose a toolkit that allows the development of interactive applications that can be interacted with via a smartphone for remote, direct-manipulation style interaction. Our toolkit uses only web technologies, making it highly platform-independent and deployable across a variety of display systems. Our toolkit provides high-level controls so that application developers can focus on the high-level interaction design of the application. These controls were based on the work by Cardoso & José [1], which identified various interaction tasks for public displays, including tasks that require a direct-manipulation paradigm.

MOBIQUITOUS 2014, December 02-05, London, Great Britain
Copyright © 2014 ICST 978-1-63190-039-6
DOI 10.4108/icst.mobiquitous.2014.258067

2. WEB-BASED REMOTE INTERACTION

Our toolkit is composed of an interaction server that relays interaction events between the mobile devices and the public display application, and a programming library. The programming library provides interaction controls (widgets) and interaction events.

To use the toolkit, application programmers include in their application a set of files that compose the programming library. The complete process of loading an application and interacting with it is depicted in Figure 1. When an application is loaded into the public display it will load all the application files including the toolkit's files (A). After loading, the toolkit will automatically establish a web socket connection to the interaction server (B). The application should then advertise some kind of connection procedure for users, for example, by displaying a QR Code. When a user scans the QR Code (C), his mobile device will load the mobile interface from the application server (D). The mobile interface files are provided by the toolkit but programmers can customize them. After loading, the mobile interface will establish a web socket connection with the interaction server (E). After these steps, interactions performed on the mobile device are transmitted via web sockets to the public display application, via high-level control events.

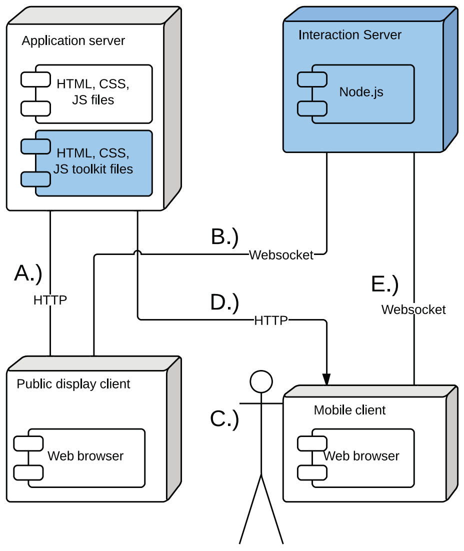
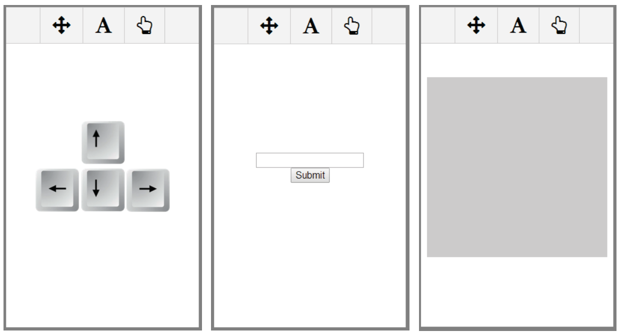


Figure 1. Architecture of the toolkit.

The toolkit provides three types of controls: joystick, text entry, and multi-touch. The graphical appearance of the controls on the mobile device can be seen in Figure 2. The default mobile interface provided by the toolkit combines all three controls in the same web view. Users can select the intended control using the buttons on the top. Developers can choose to use only one control or another, by manually deleting the graphical views of the other controls.



Joystick control Text control Multi-touch control

Figure 2. Preliminary version of the graphical representation of the controls on the mobile device.

When users interact with the controls on the mobile device, interaction events are sent to the application via the web socket connection. On the application side, our toolkit receives these events and triggers the appropriate callbacks on the application code. For example, the joystick control triggers a *joystick(userid, direction)* event callback.

Currently, we assume that the user's mobile device, the interaction server, and the public display client are all connected to the same local network. To further simplify the development of applications, we also require that the interaction server run on the same machine as the public display client. This way, applications can simply connect to *localhost* to contact the interaction server.

In the current implementation, the user decides which control to use on the mobile interface at any given time, but we plan to study different alternatives such as allowing the display application to control the mobile interface, or a hybrid approach where the display application highlights the intended control only.

We have conducted a preliminary evaluation of the toolkit by asking a group of three programming students to use it to create public display applications. Through the development of these apps (hangman game, car racing, and tetris) we were able to gather early feedback about the toolkit, its features, and implementation. Feedback from this evaluation included the addition of new controls that allow the use of mobile sensors, a better coding structure of the toolkit, and more flexibility in using the toolkit's files for the mobile interface.

3. RELATED WORK

A number of programming toolkits for interactive public displays have begun to emerge in the last years. The PureWidgets toolkit [2] provides various widgets that can be incorporated into public display applications and activated through various input mechanisms (SMS, QR Code, touch, etc.). However it does not provide any direct-manipulation control for real-time interaction. Gehring et al. [3] proposes a framework for direct-manipulation multi-touch interaction using a smartphone that captures a live video feed of the display allowing users to touch the image to send multi-touch events to the display using the TUIO protocol [5]. Our toolkit is different in that it does not require users to be constantly pointing their devices to the public display in order to capture a live video feed, and it provides higher-level controls in addition to the multi-touch cursor events. SenScreen [7] is a toolkit for sensor-enabled display networks, where a server mediates the access to the sensor data and provides higher-level APIs to the sensors. SenScreen focuses on sensors that are part of the display infrastructure such as Kinect devices, and on gesture interactions. Our toolkit is focused on user-owned devices for touch-based remote interaction with public displays.

4. CONCLUSIONS

We are developing a web-based toolkit for remote, direct manipulation interaction with public display applications. This toolkit is intended to reduce the development effort of creating these applications by providing high-level, ready to use widgets that trigger interaction events in the application. The current implementation supports basic controls such as joystick, text input, and multi-touch cursor events. We have planned to incorporate other controls that allow the use of mobile device sensors such as accelerometer and compass data.

5. ACKNOWLEDGMENTS

This paper was financially supported by the Foundation for Science and Technology — FCT — in the scope of project PEst-OE/EAT/UI0622/2014.

6. REFERENCES

[1] Cardoso, J.C.S. and José, R. 2014. Interaction Tasks and Controls for Public Display Applications. *Advances in Human-Computer Interaction*. 2014, (2014), 1–17.

[2] Cardoso, J.C.S. and José, R. 2012. PuReWidgets: a programming toolkit for interactive public display applications. *Proceedings of the 4th ACM SIGCHI symposium on Engineering interactive computing systems - EICS '12* (New York, NY, USA, Denmark, Jun. 2012), 51.

[3] Gehring, S. et al. 2012. Towards Universal , Direct Remote Interaction with Distant Public Displays. *PPD'12: Workshop on infrastructure and design challanges of coupled display visual interfaces* (2012), 14–17.

[4] José, R. et al. 2008. Instant Places: Using Bluetooth for Situated Interaction in Public Displays. *IEEE Pervasive Computing*. 7, 4 (Oct. 2008), 52–57.

[5] Kaltenbrunner, M. et al. 2005. TUIO - A Protocol for Table-Top Tangible User Interfaces. *Proceedings of the 6th International Workshop on Gesture in Human-Computer Interaction and Simulation (GW 2005)* (Vannes, France, 2005).

[6] Martin, K. et al. 2006. Engaging with a situated display via picture messaging. *CHI '06 extended abstracts on Human factors in computing systems - CHI '06* (New York, New York, USA, 2006), 1079.

[7] Schneegass, S. and Alt, F. 2014. SenScreen: A Toolkit for Supporting Sensor-enabled Multi-Display Networks. *Proceedings of The International Symposium on Pervasive Displays - PerDis '14* (New York, New York, USA, Jun. 2014), 92–97.

[8] Toye, E. et al. 2005. Using smart phones to access site-specific services. *IEEE Pervasive Computing*. 4, 2 (2005), 60–66.