# TouchDevelop — App Development on Mobile Devices

Nikolai Tillmann, Michal Moskal, Jonathan de Halleux,
Manuel Fahndrich, Sebastian Burckhardt
Microsoft Research
One Microsoft Way
Redmond WA, USA
{nikolait,micmo,jhalleux,maf,sburckha}@microsoft.com

## ABSTRACT

Mobile devices are becoming the prevalent computing platform for most people. TouchDevelop is a new mobile development environment that enables anyone with a Windows Phone to create new apps directly on the smartphone, without a PC or a traditional keyboard. At the core is a new mobile programming language and editor that was designed with the touchscreen as the only input device in mind. Programs written in TouchDevelop can leverage all phone sensors such as GPS, cameras, accelerometer, gyroscope, and stored personal data such as contacts, songs, pictures. Thousands of programs have already been written and published with TouchDevelop.

## Categories and Subject Descriptors

D.2.3 [**Software Engineering**]: Program editors; D.2.3 [**Software Engineering**]: Structured programming; D.2.6 [**Programming Environments/Construction Tools**]: Integrated environments; D.2.11 [**Software Architectures**]: Languages

## General Terms

Human Factors, Design, Languages

## Keywords

Evolution, Mobile Devices, Touchscreen, Cloud,
Type inference

## 1. INTRODUCTION

Traditionally, software applications for mobile devices such as smartphones have been developed on a PC tethered to the mobile device for testing. We present TouchDevelop [1], a novel mobile application creation environment that runs directly on a Windows Phone. No PC is required to create mobile applications. TouchDevelop comes with a typed, structured programming language that is built around the idea of only using a touchscreen as the input device to author code; built-in primitives make it easy to access the rich sensor data available on a mobile device.

Many mobile applications are rather small, and it is conceivable that they could be written on a mobile device. In fact, there are several common issues in the traditional mobile application development model that on-device development with TouchDevelop alleviates, such as sensor handling and complexity of testing, which is difficult to perform off-device in an emulator [2].

Mobile development environments are becoming popular on all major platforms. Codea[1] is a programming environment that runs exclusively on iPad; it is based on entering code via a traditional (soft) keyboard, and its APIs are mainly geared towards games. AIDE[2] enables general-purpose app development directly on Android; it is also based on entering Java-like code via a (soft) keyboard.

```
action ::= frobnicate | go draw line | ...
global ::= phone | math | maps | ok button | chat roster | ...
property ::= vibrate | sin | add pushpin | disable | add | ...
op ::= + | − | / | ∗ | <= | >= | > | < | = | != | and | or | ||
expr ::= localvar | global | string | number | true | false
     | − expr | not expr | expr op expr
     | action parameters | expr −> property parameters
block ::= stmt | block stmt
stmt ::= expr | locals := expr | do nothing
     | if expr then block else block
     | while expr do block | foreach local in expr do block
     | for 0 <= local < expr do block
type ::= Number | String | Song | Nothing | Phone | ...
formals ::= local : type | local : type , formals
action-def ::= action action ( formals-opt ) returns-opt block
```

**Figure 1: Excerpt of abstract syntax of TouchDevelop language**

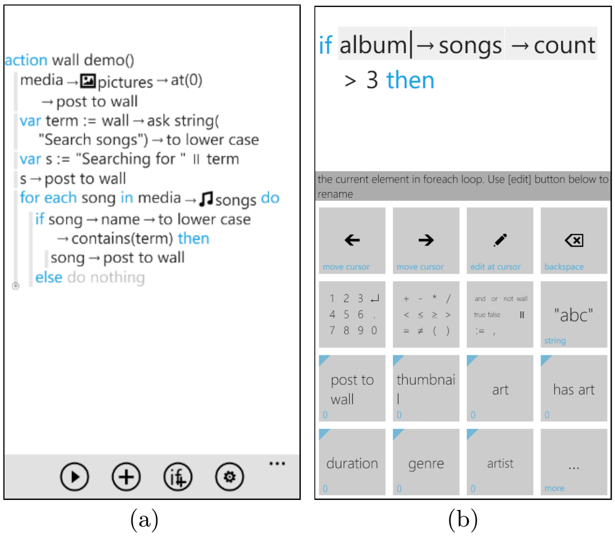
## 2. MOBILE DEVELOPMENT ENVIRONMENT

At the core of TouchDevelop is a typed, structured programming language that is built around the idea of using only a touchscreen as the input device to author code. The abstract syntax, typing rules, and basic semantics of the language are fairly similar to corresponding fragments of mainstream programming languages such as Java or C#. Our

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.
*SIGSOFT'12/FSE-20,* November 11–16, 2012, Cary, North Carolina, USA.
Copyright 2012 ACM 978-1-4503-1614-9/12/11 ...$15.00.

[1] http://twolivesleft.com/Codea/

[2] http://play.google.com/store/apps/details?id=com.aide.ui



```
action wall demo()
  media → pictures → at(0)
      → post to wall
  var term := wall → ask string(
      "Search songs") → to lower case
  var s := "Searching for " || term
  s → post to wall
  for each song in media → songs do
    if song → name → to lower case
        → contains(term) then
      song → post to wall
    else do nothing
```

```
if album| → songs → count
   > 3 then
```

the current element in foreach loop. Use [edit] button below to rename

**Figure 2: Two screenshots illustrating the TouchDevelop programming environment on mobile devices.**

language mixes imperative, object-oriented, and functional aspects. The imperative parts are most visible: assignment statements can update local variables and the state of global objects. The language has a notion of objects in order to facilitate code autocompletion by the editor — properties of objects are an easily accessible and intuitive concept. Functional features come in form of a query language. A program consists of a set of actions; each action contains statements. Figure 1 gives an overview of the abstract syntax of statements and expressions that can appear in actions. A graphical editor allows the definition of custom structured data types such as records, objects, and tables.

Instead of a traditional text editor, TouchDevelop employs a semi-structured code editor. The motivation for structured editing is not to entirely eliminate the possibility of syntax errors, but instead to ensure that every navigation task between different syntax elements can be easily achieved by tapping on a finger-sized user interface element on the screen. To determine which user interface elements are shown on the limited screen space, the editor leverages inferred types, and it mines previously written programs to provide highly predictive auto-completion suggestions to the user. In effect, the user only has to choose from lists of statement kinds and expression tokens instead of entering statements and expressions character by character.

In addition to the basic programming functionality (language, interpreter, editor), TouchDevelop provides a social experience: users can publish and download programs, discuss them by writing comments, publish screenshots, and give hearts when they like programs. A central program repository in the cloud manages all programs and analyzes them for privacy concerns and code duplication.

Figure 2 shows different aspects of the TouchDevelop code editors, including a listing of a complete program. Each screenshot captures the entire screen of a smartphone. Note that all surface areas that act as buttons are large enough to be tapped on quite precisely by a big finger. The left screenshot (a) shows a list of statements. The user can select, add or remove a statement by tapping somewhere in a statement line. It is not possible to make syntactical mistakes at the statement level. When editing a statement, the view transitions to the expression editor to edit the expression associated with the statement. The right screenshot (b) shows the expression editor, which is implemented similar to a rich calculator, and can be seen as a domain-specific keyboard. Individual built-in syntactic primitives such as numbers and operators can be accessed via nested keypads, opened by first tapping on the corresponding summary button. Other semantic procedure and function calls can be inserted by eight quick buttons, which are populated based on the type of the current scope, and historical usage information.

## 3. EXPERIENCE

After nine months, more than 6000 games and applications have been written and published by users of TouchDevelop, using the touchscreen as the only input device. TouchDevelop has been used in middle- and high-school classes, and for the entire duration of a general introduction to computer science class at UCSC.

Many games written by users leverage the accelerometer of the phone as a controller, measuring how the phone is tilted, including a fully-featured games that mimic Tetris, Breakout, Doodle Jump, and more. Other applications are useful tools and utilities, including a program that queries the calendar on the phone and automatically connects to any currently scheduled online meeting, a program that records the GPS locations of bird sightings where the table of birds is queried via a web service, a Siri-like application that connects the microphone, a speech recognition engine, and a search engine via web service calls, and many more.

## 4. AVAILABILITY

TouchDevelop is available as a free application on the Windows Phone Marketplace.[3] The TouchDevelop website[4] shows all games and applications published by TouchDevelop users.

## 5. CONCLUSION

In the foreseeable future, incredibly powerful and easy-to-use smartphones will be the first (and possibly only) computing devices, which virtually all people will own and carry with them at all times. We presented TouchDevelop, an application creation environment that embraces the form factor and input capabilities of such mobile devices.

## 6. REFERENCES

[1] N. Tillmann, M. Moskal, J. de Halleux, and M. Fahndrich. Touchdevelop: programming cloud-connected mobile devices via touchscreen. In *Proc. of SPLASH*, ONWARD '11, pages 49–60, New York, NY, USA, 2011. ACM.

[2] A. I. Wasserman. Software engineering issues for mobile application development. In *Proc. of FSE/SDP*, FoSER '10, pages 397–400, New York, NY, USA, 2010. ACM.

[3] http://windowsphone.com/s?appId=fe08ccec-a360-e011-81d2-78e7d1fa76f8

[4] http://touchdevelop.com/

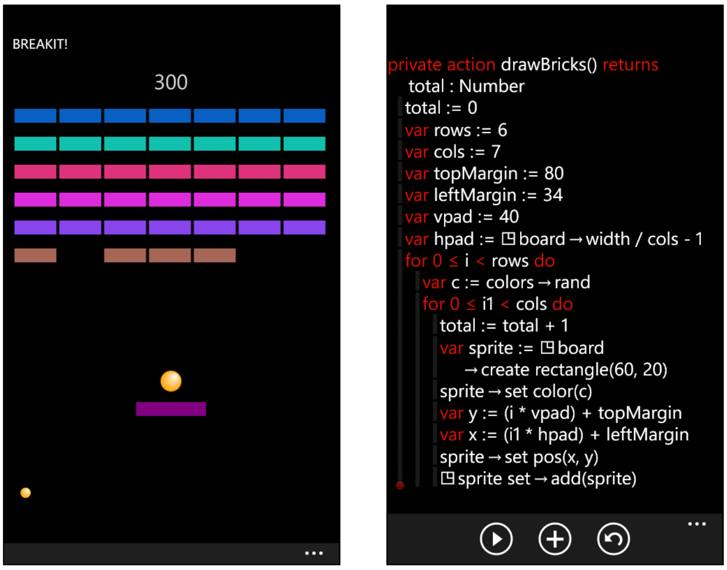
# APPENDIX

The demonstration starts with a general walkthrough of the programming environment, showing some of the programs written and published by other users. Figure 3 shows the code and a program in action.

The main portion of the demonstration consists of creating and publishing a mobile application with the following functionality. It has an event handler that triggers when the phone is shaken (Figure 4). The event handler then chooses a picture from the user's picture library (Figure 5, 6), applies an effect (here, tinting in a random color), and posts the transformed picture on the screen. Figure 7 shows the code statements and illustrates their execution. Finally, we will publish the program (Figure 8) to the TouchDevelop script bazaar in the cloud. A video illustrating the coding part of the demonstration is available online.[5]

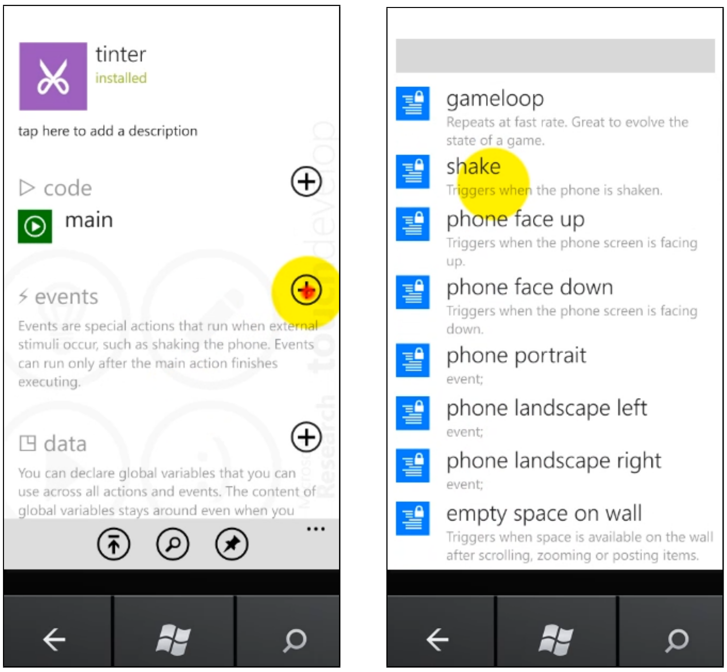
Screenshots may show a yellow circle that illustrates where the user taps.

[5] http://www.touchdevelop.com/doc?path=videos%252fendtoend.mp4



```
private action drawBricks() returns
  total : Number
 total := 0
 var rows := 6
 var cols := 7
 var topMargin := 80
 var leftMargin := 34
 var vpad := 40
 var hpad := ◫board→width / cols - 1
 for 0 ≤ i < rows do
   var c := colors→rand
   for 0 ≤ i1 < cols do
     total := total + 1
     var sprite := ◫board
        →create rectangle(60, 20)
     sprite→set color(c)
     var y := (i * vpad) + topMargin
     var x := (i1 * hpad) + leftMargin
     sprite→set pos(x, y)
     ◫sprite set→add(sprite)
```

**Figure 3: A break-out program that uses the accelerometer to move the paddle. The code shows how the grid of bricks is created.**



tinter
installed

tap here to add a description

code

main

events

Events are special actions that run when external stimuli occur, such as shaking the phone. Events can run only after the main action finishes executing.

data

You can declare global variables that you can use across all actions and events. The content of global variables stays around even when you

gameloop
Repeats at fast rate. Great to evolve the state of a game.

shake
Triggers when the phone is shaken.

phone face up
Triggers when the phone screen is facing up.

phone face down
Triggers when the phone screen is facing down.

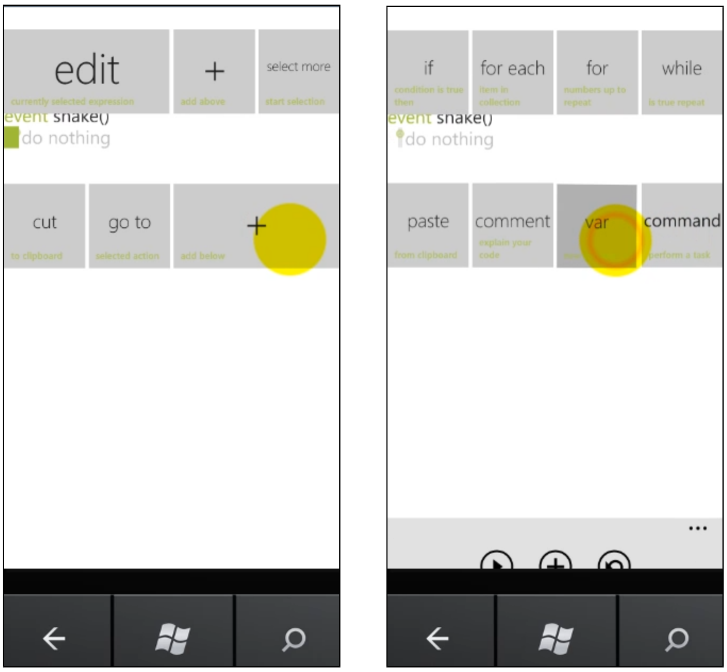
phone portrait
event;

phone landscape left
event;

phone landscape right
event;

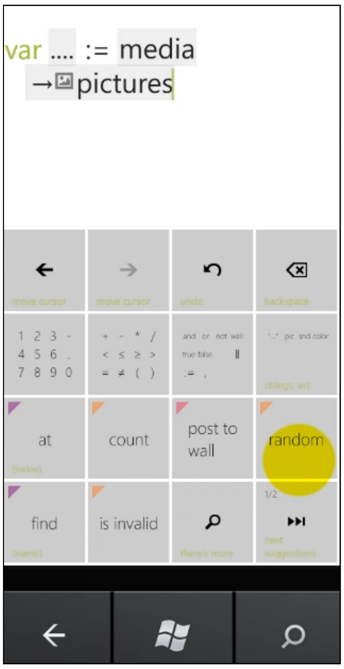
empty space on wall
Triggers when space is available on the wall after scrolling, zooming or posting items.

**Figure 4: A new event is added by first tapping on the '+' next to events, and then selecting which kind of event to subscribe to.**



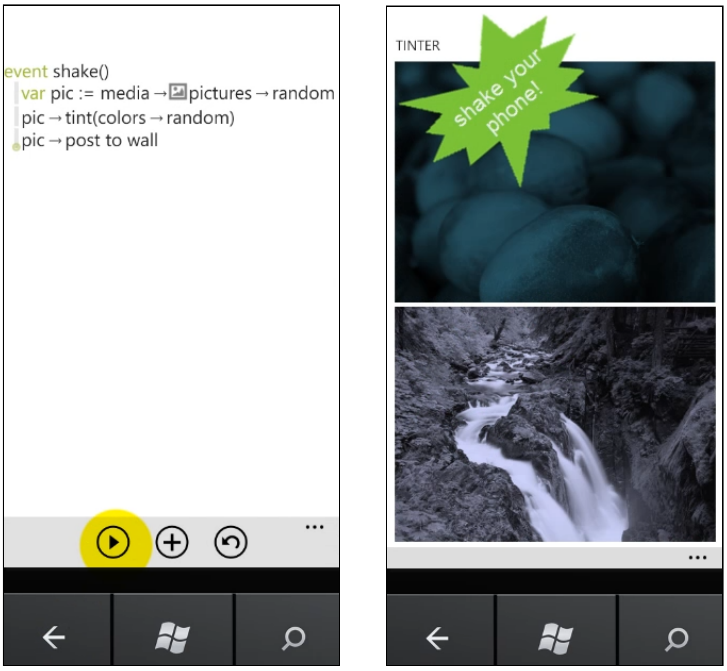
```
event snake()
do nothing
```

**Figure 5: The code of an event handler is written in a imperative, procedural style. A new statement is inserted by first tapping on a line, then selecting '+', and finally selecting which kind of control-flow construct or simple statement should be added.**



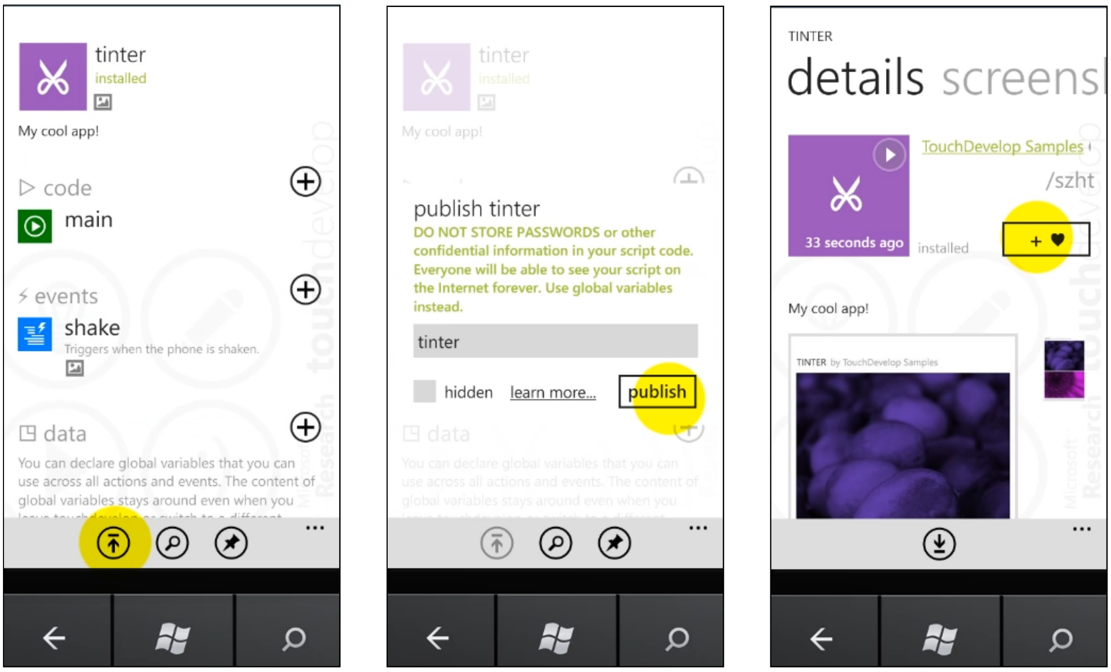
```
var .... := media
  →pictures
```

**Figure 6: Expressions are inserted in a token-based, calculator-like editor. Besides integers, arithmetic and Boolean operators, the lower half of the calculator buttons offer commonly-used properties.**



```
event shake()
  var pic := media→pictures→random
  pic→tint(colors→random)
  pic→post to wall
```

Figure 7: Back in the event view, this screen shows the entire program that we'll create in the demo. It can be run via a 'run' button. While running, every time the phone is shaken, a random picture is picked from the media library, tinted, and posted to the screen.



Triggers when the phone is shaken.

You can declare global variables that you can use across all actions and events. The content of global variables stays around even when you

DO NOT STORE PASSWORDS or other confidential information in your script code. Everyone will be able to see your script on the Internet forever. Use global variables instead.

tinter

□ hidden learn more... publish

Figure 8: A program can be shared by publishing it to the TouchDevelop script bazaar in the cloud. Publishing involves only a few taps. After also uploading a screenshot, it appears in the public script bazaar.