Microsoft

Microsoft

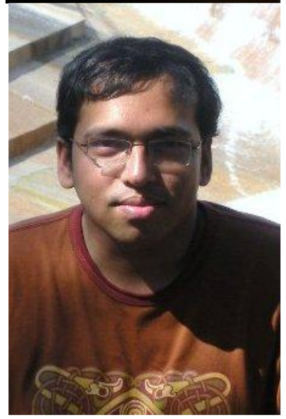
Research Faculty Summit 2012

ADVANCING THE STATE OF THE ART

An Experiment in Developing Small Mobile Phone Applications: Comparing On-Phone to Off-Phone Development

Tuan A. Nguyen,
Sarker T.A. Rumee,
Christoph Csallner
(University of Texas at Arlington)

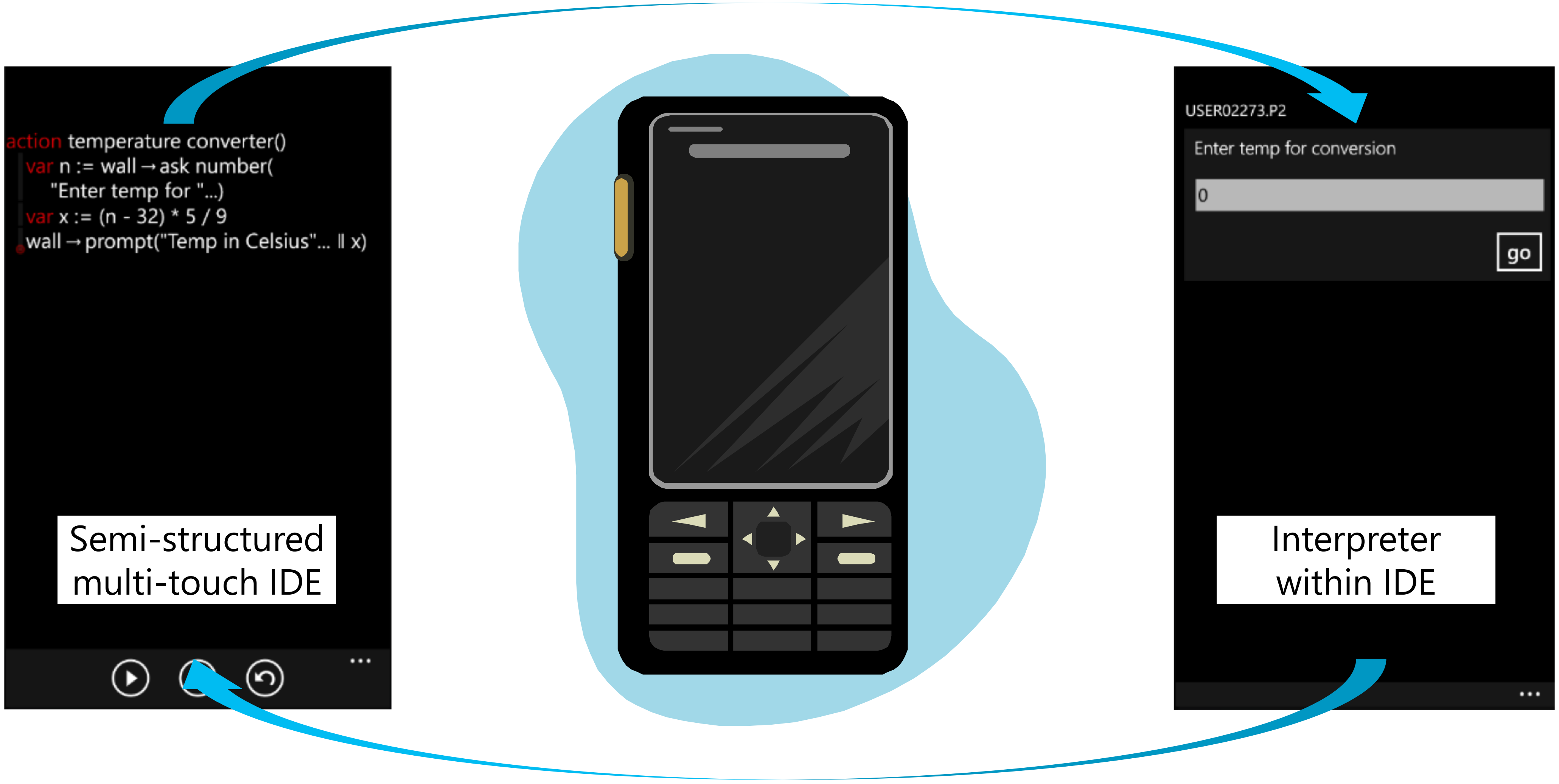
Nikolai Tillmann
(Microsoft Research)



This material is based upon work supported by the National Science Foundation under Grants No. 1017305 and 1117369. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

Monday, July 16

New Development Style: On-Phone



```
action temperature converter()
  var n := wall→ask number(
      "Enter temp for "...)
  var x := (n - 32) * 5 / 9
  wall→prompt("Temp in Celsius"... ∥ x)
```

Semi-structured multi-touch IDE

USER02273.P2

Enter temp for conversion

0

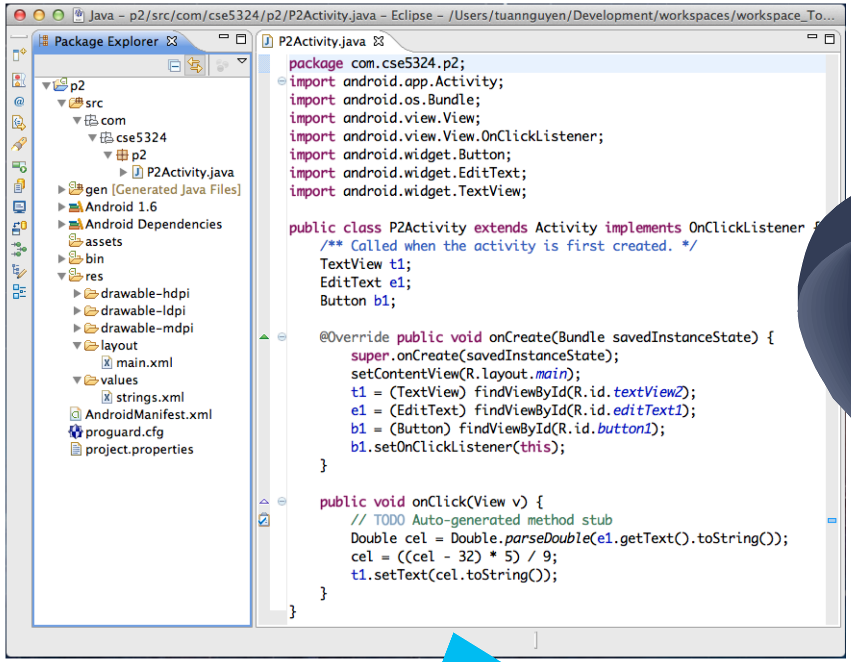
go

Interpreter within IDE

Traditional Development Style: Off-Phone

Android as an example, but Windows Phone, iOS, etc. work similarly

Standard IDE such as Eclipse, Visual Studio

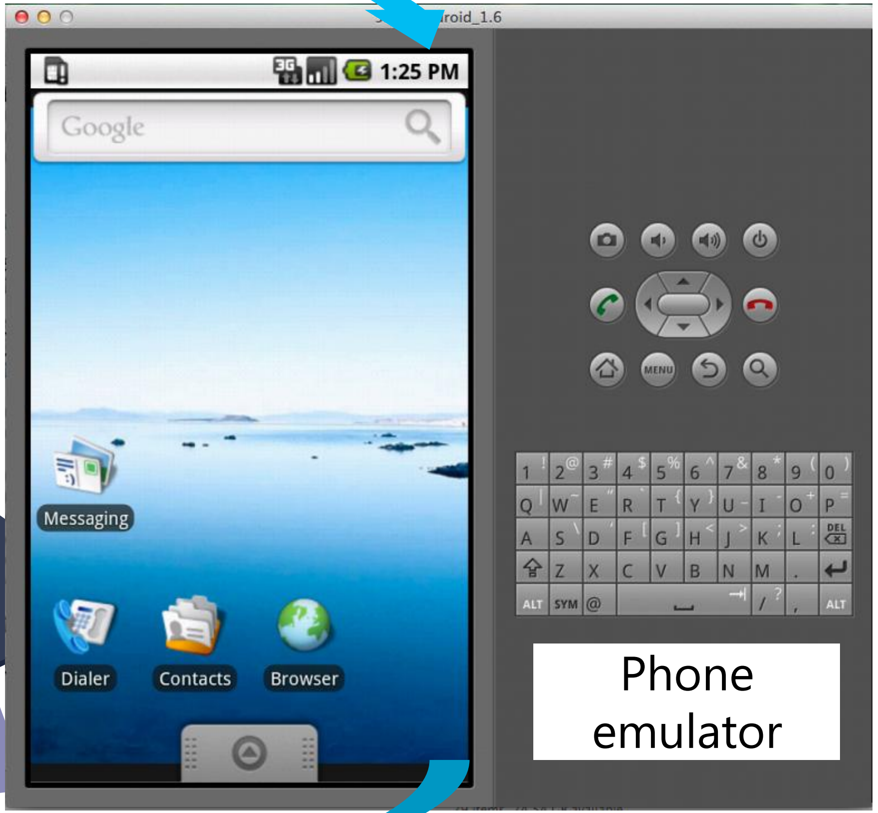

Java – p2/src/com/cse5324/p2/P2Activity.java – Eclipse – /Users/tuannguyen/Development/workspaces/workspace_To...

Package Explorer

```
▼ p2
  ▼ src
    ▼ com
      ▼ cse5324
        ▼ p2
          ▶ P2Activity.java
  ▶ gen [Generated Java Files]
  ▶ Android 1.6
  ▶ Android Dependencies
    assets
  ▶ bin
  ▼ res
    ▶ drawable-hdpi
    ▶ drawable-ldpi
    ▶ drawable-mdpi
    ▼ layout
        main.xml
    ▼ values
        strings.xml
    AndroidManifest.xml
    proguard.cfg
    project.properties
```

P2Activity.java

```java
package com.cse5324.p2;
import android.app.Activity;
import android.os.Bundle;
import android.view.View;
import android.view.View.OnClickListener;
import android.widget.Button;
import android.widget.EditText;
import android.widget.TextView;

public class P2Activity extends Activity implements OnClickListener {
    /** Called when the activity is first created. */
    TextView t1;
    EditText e1;
    Button b1;

    @Override public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.main);
        t1 = (TextView) findViewById(R.id.textView2);
        e1 = (EditText) findViewById(R.id.editText1);
        b1 = (Button) findViewById(R.id.button1);
        b1.setOnClickListener(this);
    }

    public void onClick(View v) {
        // TODO Auto-generated method stub
        Double cel = Double.parseDouble(e1.getText().toString());
        cel = ((cel - 32) * 5) / 9;
        t1.setText(cel.toString());
    }
}
```



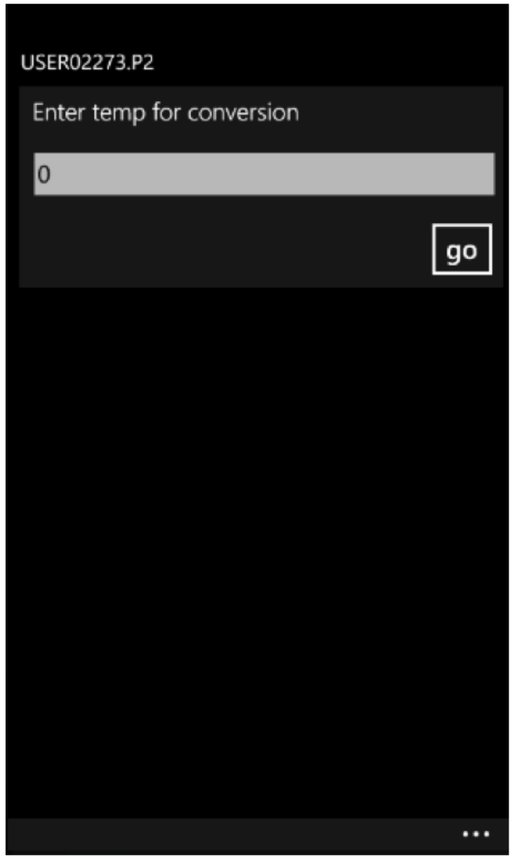
Phone emulator

Resulting Apps May Look Very Similar

Example: Convert degrees Fahrenheit to degrees Celsius

TouchDevelop

Android

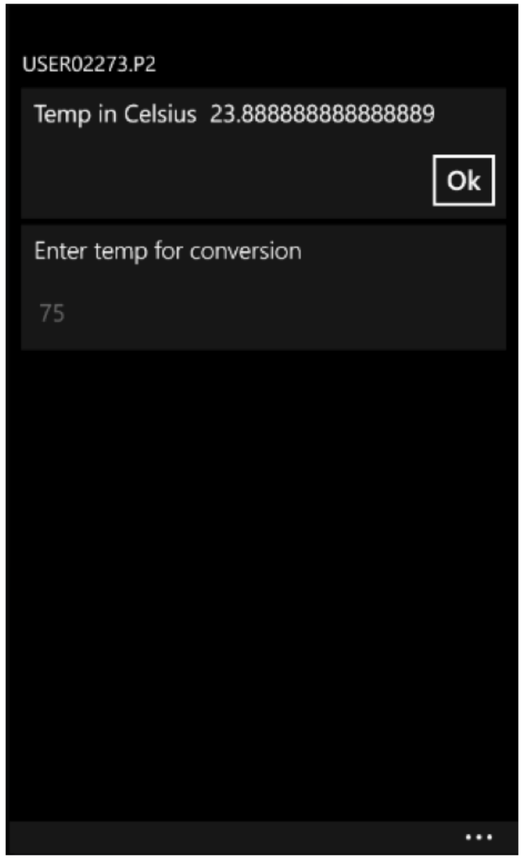


USER02273.P2

Enter temp for conversion

0

go



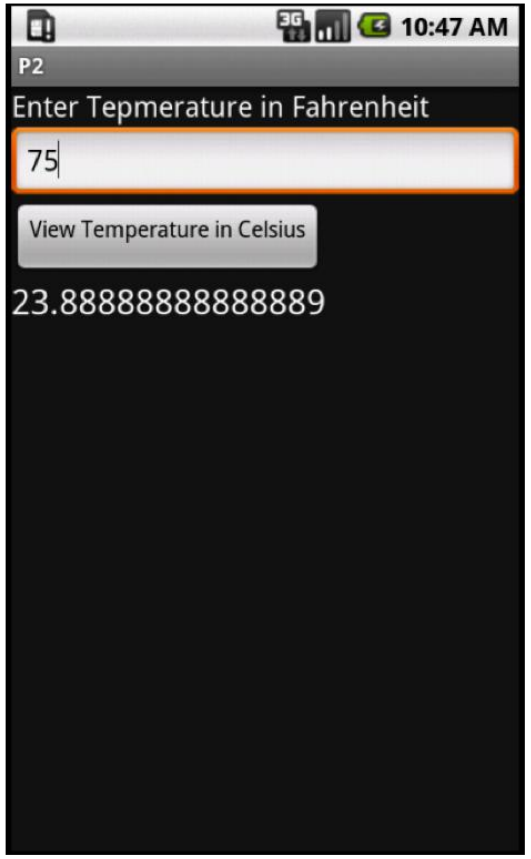
USER02273.P2

Temp in Celsius 23.888888888888889

Ok

Enter temp for conversion

75



10:47 AM

P2

Enter Tepmerature in Fahrenheit

75

View Temperature in Celsius

23.88888888888889

On-Phone vs. Traditional Off-Phone Development

Research questions (RQ), expectations (E), and hypotheses (H)

RQ1: How large are TouchDevelop apps?

E1: Tiny phone screen → expect most TouchDevelop apps to be small

H1: TouchDevelop Apps Are Small

RQ2: For given task: TouchDevelop-LOC vs. Android-LOC

E2: TouchDevelop specialized, assumes and hides details → expect TouchDevelop apps to be smaller

H2: TouchDevelop-LOC < Android-LOC

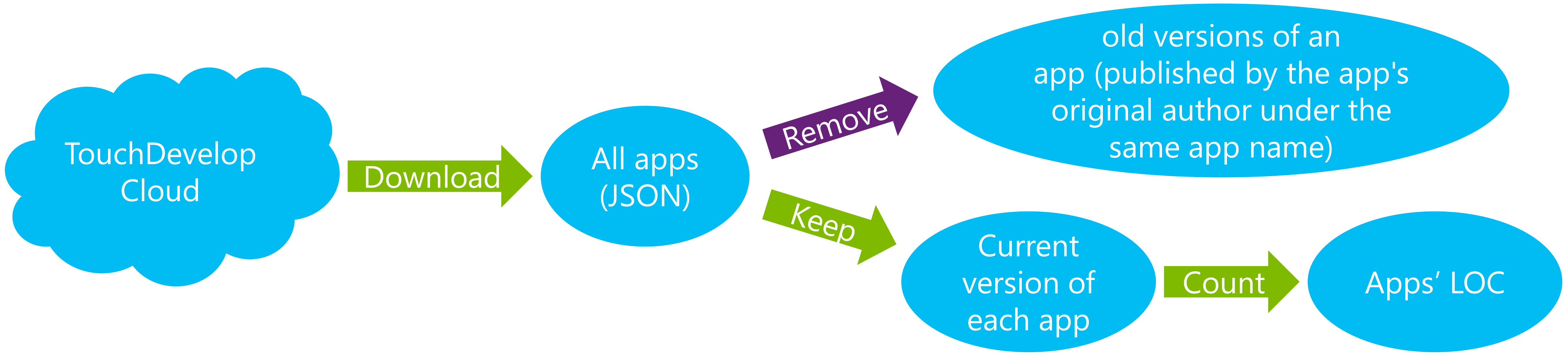
RQ3: Programmer productivity: TouchDevelop vs. Android

E3: Tiny phone screen, no keyboard, no mouse → expect TouchDevelop programmers to be less productive

H3: TouchDevelop-Productivity < Android-Productivity

RQ1: How Large Are TouchDevelop apps?

Count LOC of all TouchDevelop apps in TouchDevelop cloud


TouchDevelop Cloud
Download
All apps (JSON)
Remove
old versions of an app (published by the app's original author under the same app name)
Keep
Current version of each app
Count
Apps' LOC

Counting TouchDevelop LOC (and Android LOC)

Normalize TouchDevelop and Android apps

Count logical source statements (LSS) *["Software Size Measurement: A Framework for Counting Source Statements" by Park. Technical report, Software Engineering Institute, Sept. 1992.]*

Do not count content of configuration files (XML, text, etc.)

RQ2 & RQ3: Experiment on Student Subjects

27 students of CSE 5324 software engineering class

- 25 MS CS + 1 MS CE + 1 MS SE, taught by Csallner, Rumee: TA
- In this class students (expect to) work on big Android team project (team = 5 students)
- Experiment conducted toward end of semester → students have some Android experience

10 Samsung Focus Windows 7 Phones from Microsoft

- University lab with > 17 lab PCs
- Randomly assigned subjects to:
  - 10 WP7 phones with pre-installed TouchDevelop v2.4.0.0 beta

Thanks to Microsoft Research Connections

  - 17 lab PCs with pre-installed Eclipse + Android SDK v 1.6

One class period

- 10 min informed consent & phone loan forms + 60 min tasks + 10 min questionnaire
- Stressed that participation does not influence grades, provided link to respective APIs
- Individual development: Can consult samples, web, no other communication except with instructor/TAs

Experiment Mechanics

Windows Phone subjects

Did not receive training in TouchDevelop

- Received link to TouchDevelop website and 2-minute intro video

Did 5-minute phone setup

- Setup wireless internet connection
- Enter assigned fresh Windows Live ID
- Download & install TouchDevelop

TouchDevelop comes with samples

Students not allowed to use another device

- Simulate phone-only development

Developed apps published to cloud by TA

Android lab PC subjects

Had taught themselves Android for class project

Did 5-min PC setup

- Download & install Android Development Tools
- Create & start virtual device (emulator)

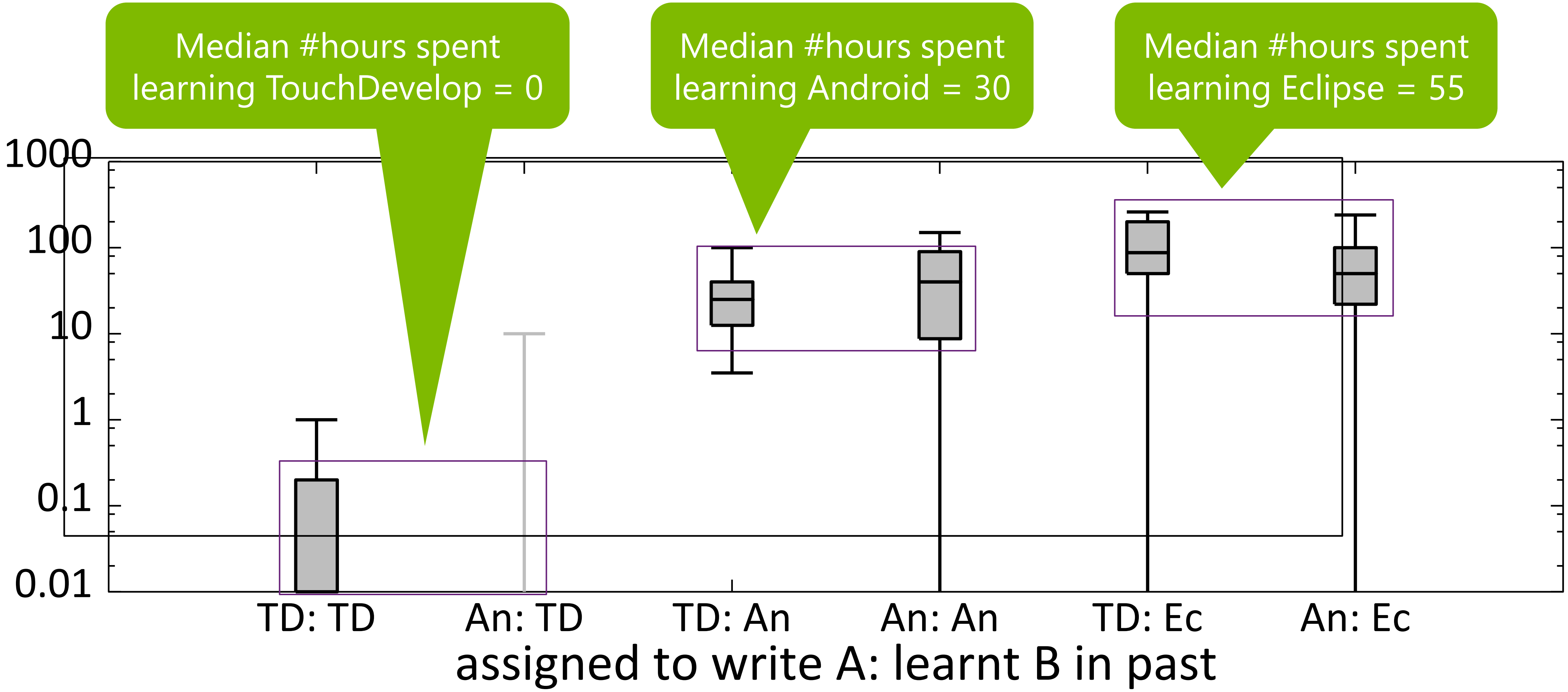
Wizard generates working “Hello World” program

E-mailed source of developed apps to TA

- 2 subjects failed to do that
  → Left with 15 Android subjects

Subjects Had Little Prior TouchDevelop Knowledge

Asked for hours spent before experiment on learning TouchDevelop (TD), Android (An), Eclipse (Ec)



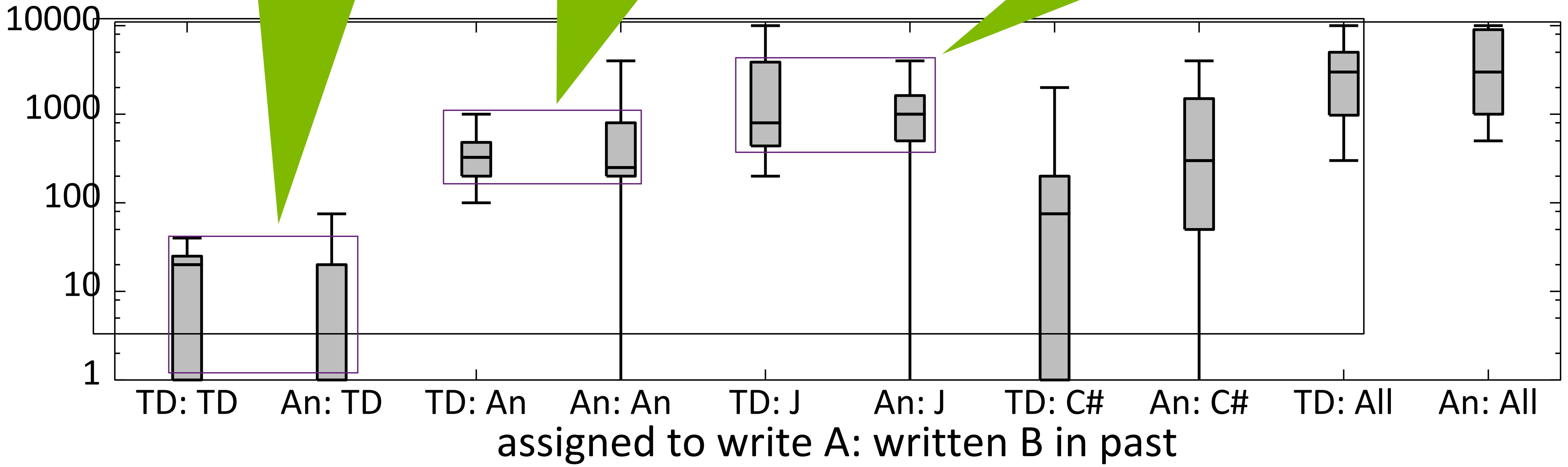
Prior LOC: TouchDevelop < Android < Java

Asked for LOC written before experiment in TouchDevelop (TD), Android (An), non-Android Java (J), C#

Median TouchDevelop LOC written before experiment = 0

Median Android LOC written before experiment = 250

Median non-Android Java LOC written before experiment = 1,000



Experiment Design: 11 Programming Tasks

(P1) Any "Hello World" program that prints "CSE 5324" on the screen.

(P2) A program that takes as input an integer number representing degrees Fahrenheit, converts it to degrees Celsius (using the Fahrenheit to Celsius conversion rule: deduct 32, multiply by 5, then divide by 9), and prints the resulting value.

(P3) A tip calculator that takes as input two integer numbers A, B from the user and prints the value of A*B/100.

(P4) A program that takes as input an integer number and prints "even" if it is an even number and "odd" if it is an odd number.

(P5) A program that takes as input a string and a character, prints "contains" if the string contains the character or else prints "not in there".

(P6) A program that takes as input a string and prints out the string with first character in uppercase.

(P7) A program that prints the system's current time as text.

(P 8) A program that asks the user for a positive integer value n and prints odd numbers between 0 and n (including n if n is odd).

(P9) A program that takes as input a string that consists of numbers separated by commas. The program should output the numbers in increasing order.

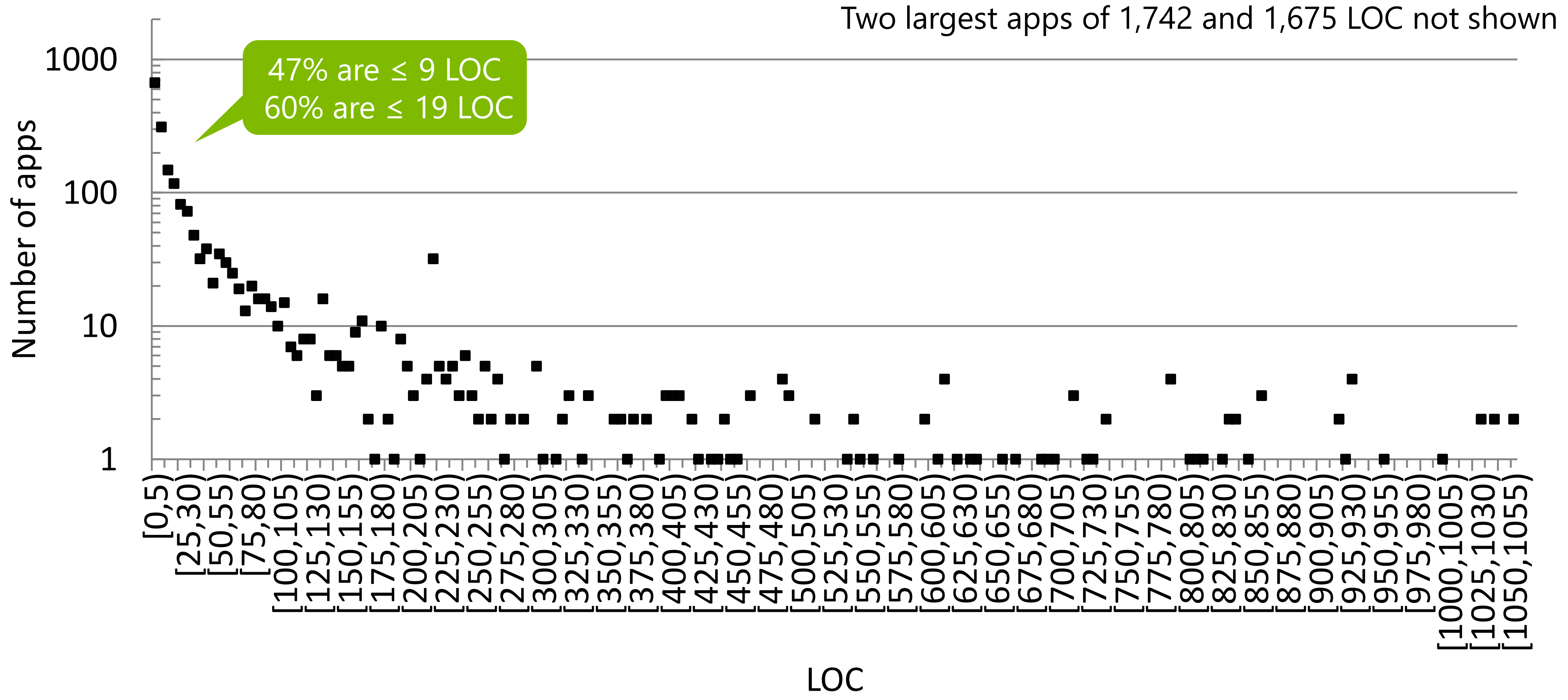
(P10) A program that draws a circle on the screen.

(P11) A program that takes two strings as input and checks if they are equal.

RQ1: How Large Are TouchDevelop apps?

Result: TouchDevelop Apps Are Small

2,081 apps in the TouchDevelop cloud (17 Feb 2012)

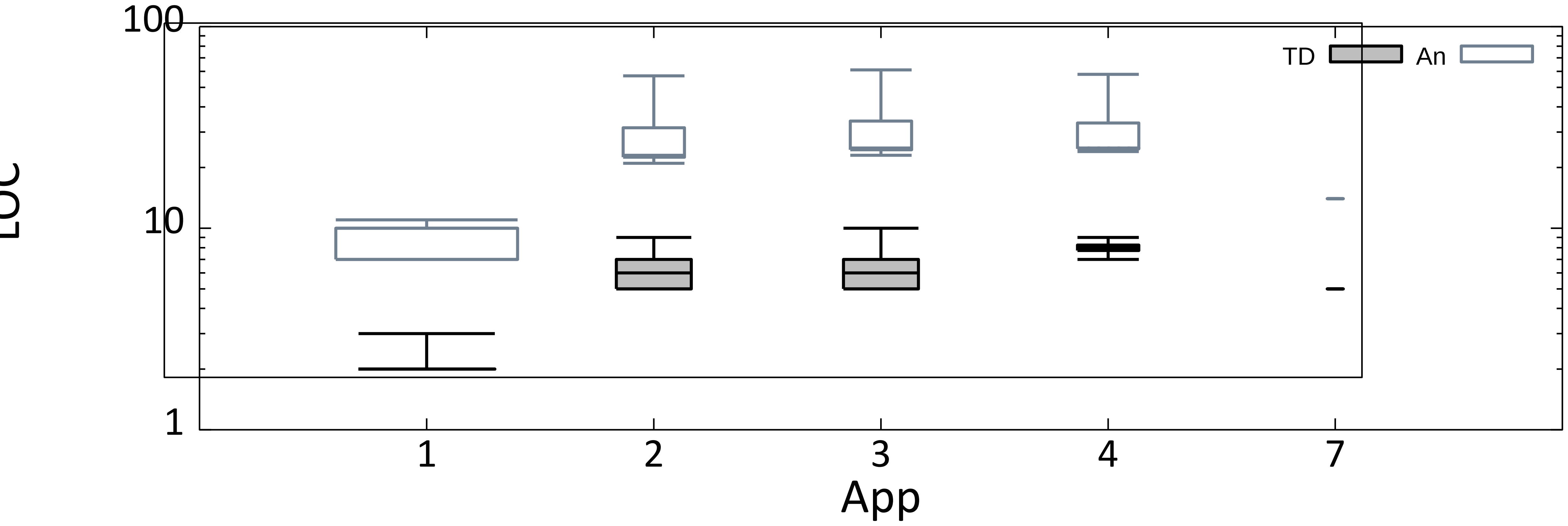


RQ2: For Given Task: TouchDevelop LOC vs. Android LOC

On Same Task: TD-LOC < Android-LOC

Correct solutions: TouchDevelop-LOC ≈ 4 Android-LOC

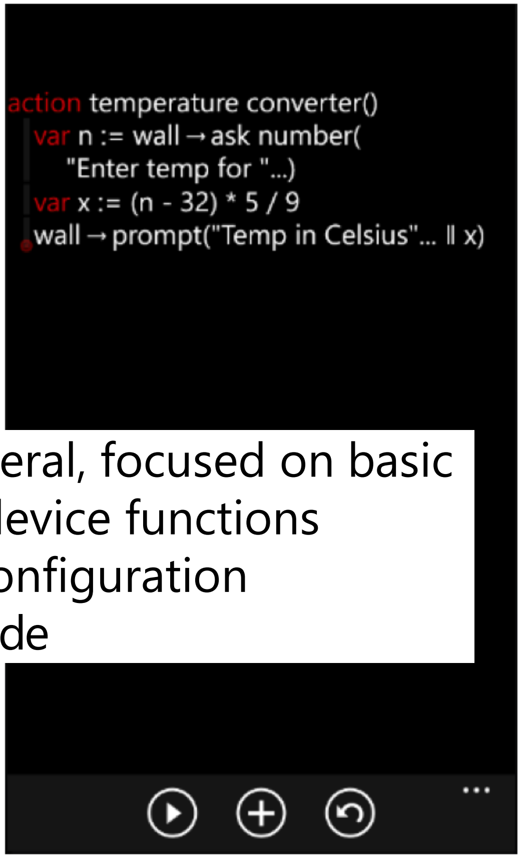
Correctness judged manually, width is proportional to #correct solutions we received



TouchDevelop App Has Less Code

Example: Convert degrees Fahrenheit to degrees Celsius

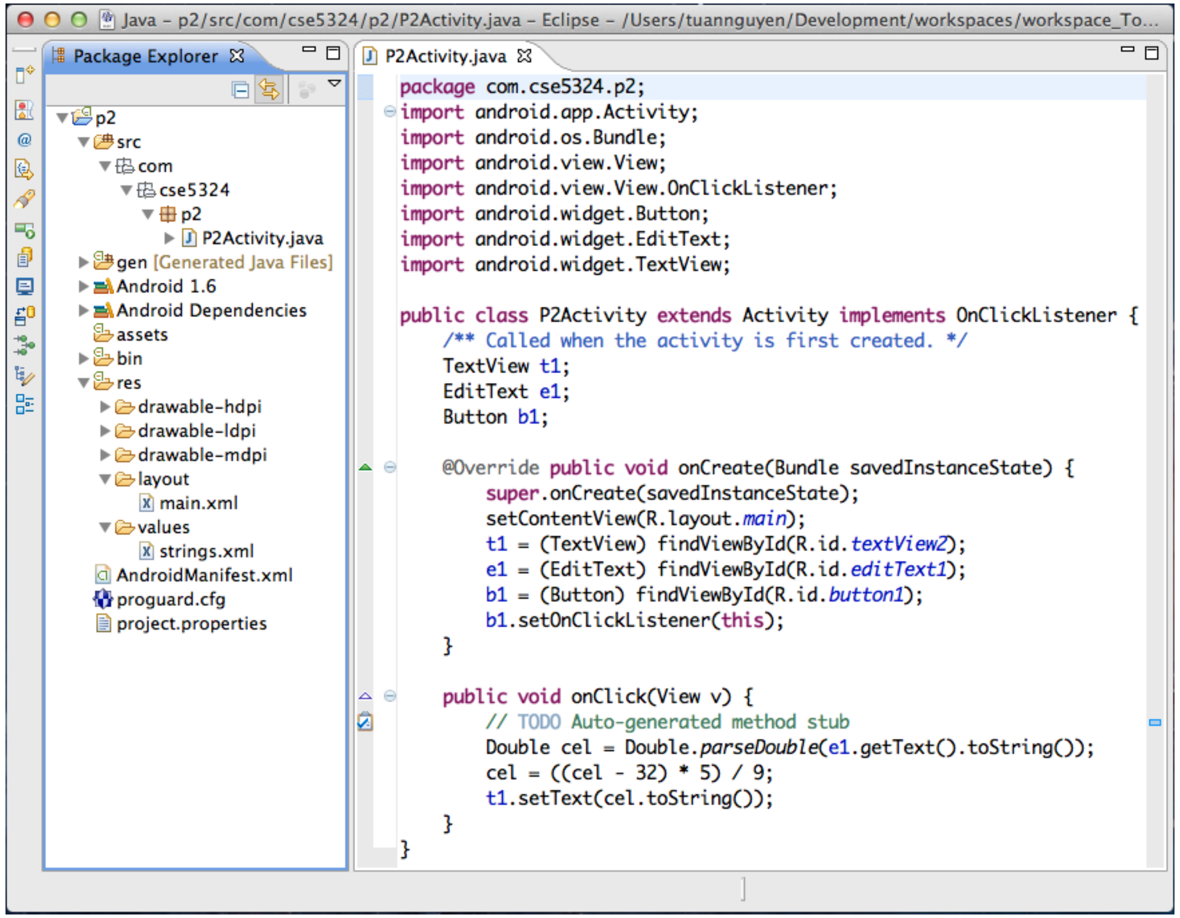
TouchDevelop



```
action temperature converter()
  var n := wall→ask number(
     "Enter temp for "...)
  var x := (n - 32) * 5 / 9
  wall→prompt("Temp in Celsius"... ∥ x)
```

Less general, focused on basic mobile device functions
+ Less configuration
+ Less code

Android



Java – p2/src/com/cse5324/p2/P2Activity.java – Eclipse – /Users/tuannguyen/Development/workspaces/workspace_To...

Package Explorer

```
p2
  src
    com
      cse5324
        p2
          P2Activity.java
  gen [Generated Java Files]
  Android 1.6
  Android Dependencies
  assets
  bin
  res
    drawable-hdpi
    drawable-ldpi
    drawable-mdpi
    layout
      main.xml
    values
      strings.xml
  AndroidManifest.xml
  proguard.cfg
  project.properties
```

P2Activity.java

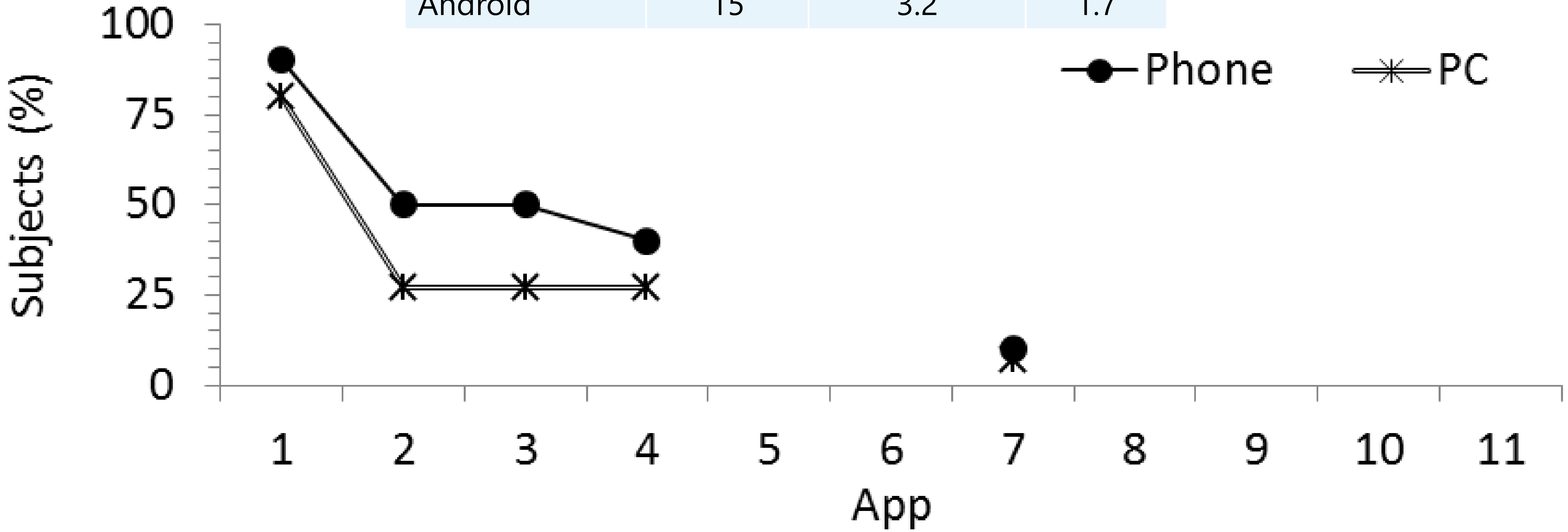
```java
package com.cse5324.p2;
import android.app.Activity;
import android.os.Bundle;
import android.view.View;
import android.view.View.OnClickListener;
import android.widget.Button;
import android.widget.EditText;
import android.widget.TextView;

public class P2Activity extends Activity implements OnClickListener {
    /** Called when the activity is first created. */
    TextView t1;
    EditText e1;
    Button b1;

    @Override public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.main);
        t1 = (TextView) findViewById(R.id.textView2);
        e1 = (EditText) findViewById(R.id.editText1);
        b1 = (Button) findViewById(R.id.button1);
        b1.setOnClickListener(this);
    }

    public void onClick(View v) {
        // TODO Auto-generated method stub
        Double cel = Double.parseDouble(e1.getText().toString());
        cel = ((cel - 32) * 5) / 9;
        t1.setText(cel.toString());
    }
}
```

RQ3: Programmer Productivity: TouchDevelop vs. Android

TouchDevelop-Productivity > Android-Productivity

TouchDevelop subjects on average started and finished more apps

System	Subjects	Apps / subject	
		Some code	Correct
TouchDevelop	10	3.7	2.4
Android	15	3.2	1.7



Why Were TD Subjects More Productive?

Possible explanations

TouchDevelop is more focused on tasks

- Less configuration and setup in TouchDevelop

Modern language features: Type-inference

- Convenient for small tasks

Semi-structured TouchDevelop IDE

- Why traditional IDEs do not have this feature?
- Traditional development may benefit from semi-structured IDEs

Threats to Validity

Our subjects are not a random sample

Study may not generalize well to novice or hobbyist programmers world wide

Subjects all UTA students

Subjects self-selected graduate software engineer course that has a large Android project

→ But subjects were not aware of this experiment

Hands-off administration of tasks

Did not instruct subjects on how to work on the given tasks

Time limits for each task may produce results that are easier to compare, but prevents subjects from switching back to earlier tasks and reusing later solutions in earlier tasks

Results may not generalize to larger programs

Limited mobile phone screen size and the limited amount of time

Designed all tasks to be simple, small, and solvable with both Android and TouchDevelop

Program size increases → TouchDevelop may need more scrolling & navigation → lower productivity

Conclusions & Future Work

Conclusions

Programmers so far have written small TouchDevelop apps

Experiment comparing on-phone to off-phone development

- Small programming tasks
- Student subjects
- Subject training: Android > TouchDevelop

→ TouchDevelop LOC < Android LOC

→ TouchDevelop productivity > Android productivity

Future Work

Why were TouchDevelop subjects more productive?

How large is the impact of the semi-structured IDE vs. the other TouchDevelop components?
E.g.: Observe programmers on semi-structured IDE vs. on un-structured version of same IDE

What happens for larger programs?

Challenging to write large programs on small screen

Are TouchDevelop maintainers also more productive?

Observe subjects as they add/change a small feature in an third-party TouchDevelop app

Are TouchDevelop testers also more productive?

How do you debug and test code on a phone?

References

Technical paper

"An experiment in developing small mobile phone applications comparing on-phone to off-phone development"
By Tuan A. Nguyen, Sarker T.A. Rumee, Christoph Csallner, and Nikolai Tillmann.
In Proc. 1st International Workshop on User Evaluation for Software Engineering Researchers (USER), June 2012, pp. 9-12.

Corpus of TouchDevelop apps, Experiment tasks & resulting TouchDevelop & Android apps, Questionnaire, tools:

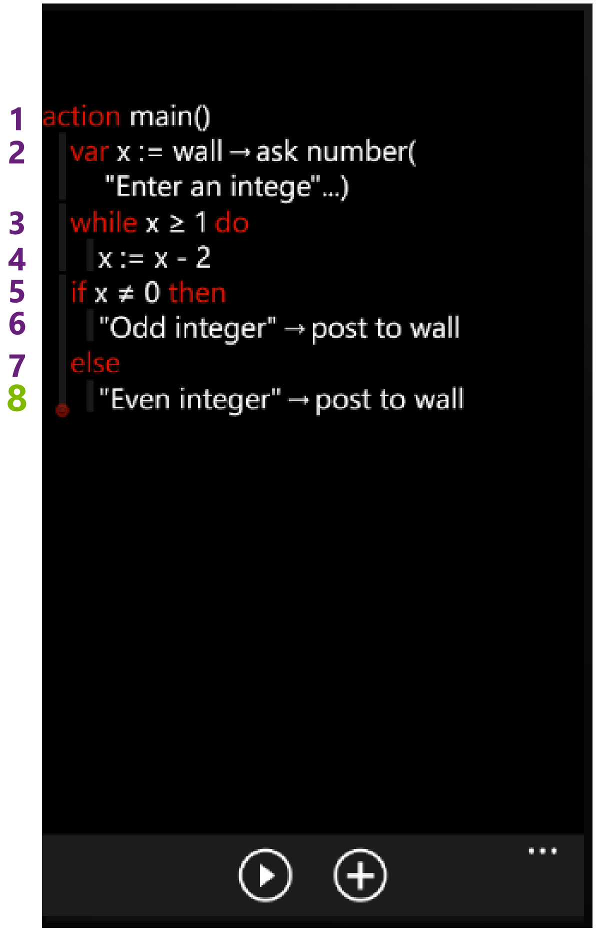
→ http://cseweb.uta.edu/~tuan/tdexp/

More Details

Counting LOC: Logical Source Statements

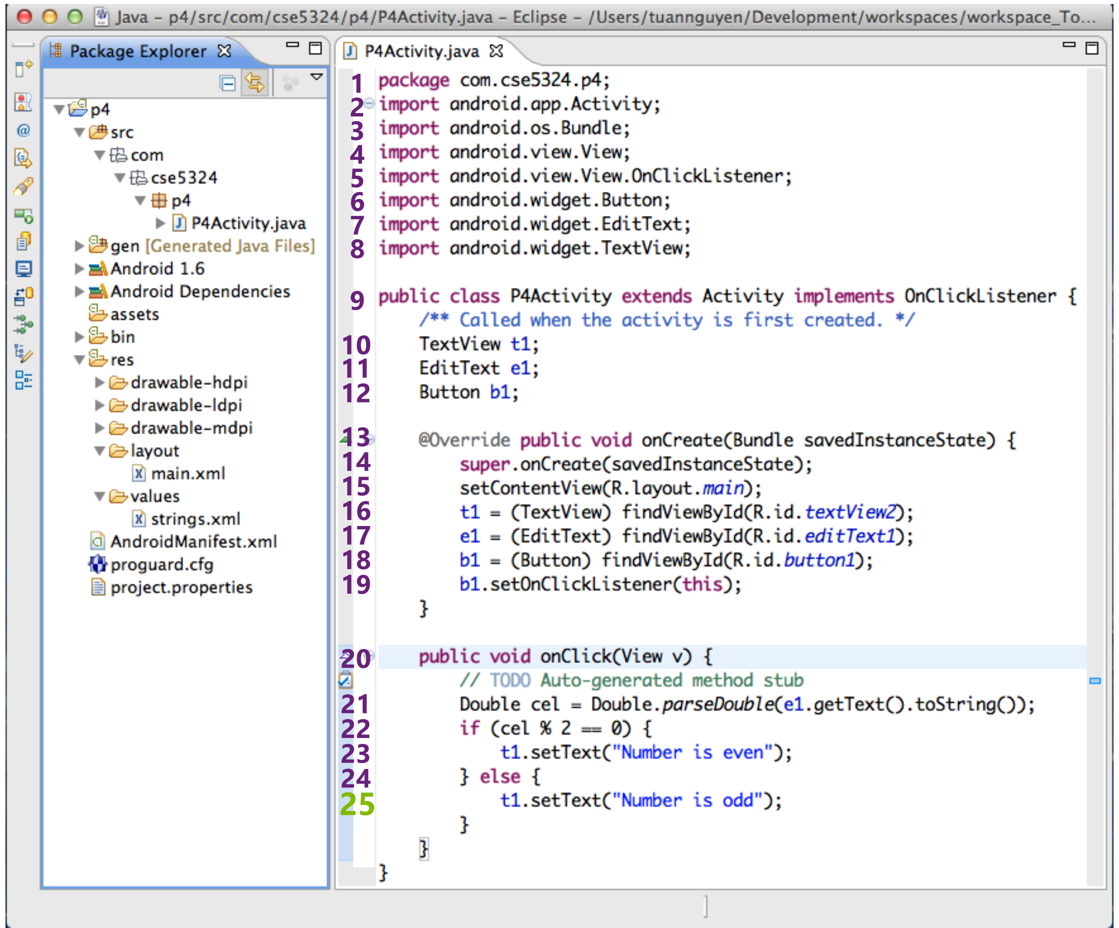
Example: Input an integer number and print "even" if it is an even number and "odd" if it is an odd number.

TouchDevelop LSS: 8



```
1 action main()
2    var x := wall → ask number(
         "Enter an intege"...)
3    while x ≥ 1 do
4       x := x - 2
5    if x ≠ 0 then
6       "Odd integer" → post to wall
7    else
8       "Even integer" → post to wall
```

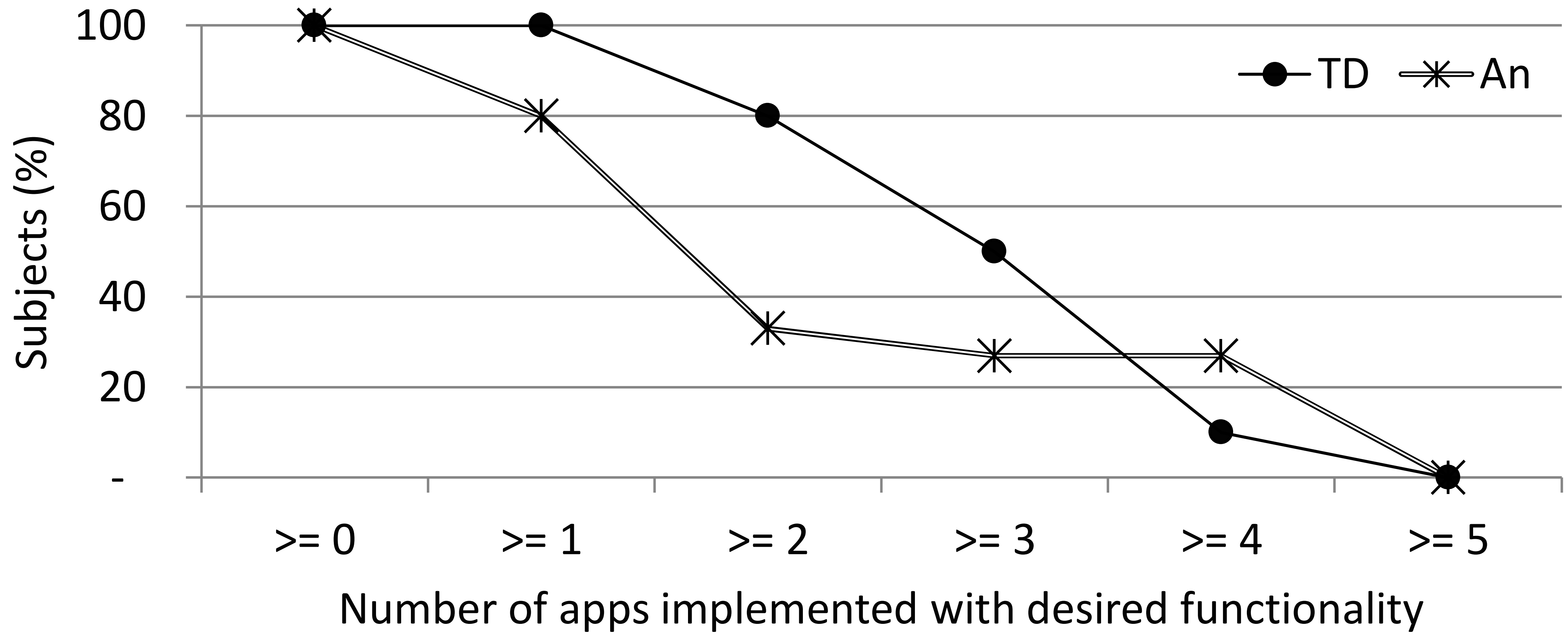
Android LSS: 25



```
1  package com.cse5324.p4;
2  import android.app.Activity;
3  import android.os.Bundle;
4  import android.view.View;
5  import android.view.View.OnClickListener;
6  import android.widget.Button;
7  import android.widget.EditText;
8  import android.widget.TextView;

9  public class P4Activity extends Activity implements OnClickListener {
       /** Called when the activity is first created. */
10     TextView t1;
11     EditText e1;
12     Button b1;

13     @Override public void onCreate(Bundle savedInstanceState) {
14         super.onCreate(savedInstanceState);
15         setContentView(R.layout.main);
16         t1 = (TextView) findViewById(R.id.textView2);
17         e1 = (EditText) findViewById(R.id.editText1);
18         b1 = (Button) findViewById(R.id.button1);
19         b1.setOnClickListener(this);
       }

20     public void onClick(View v) {
           // TODO Auto-generated method stub
21         Double cel = Double.parseDouble(e1.getText().toString());
22         if (cel % 2 == 0) {
23             t1.setText("Number is even");
24         } else {
25             t1.setText("Number is odd");
           }
       }
}
```

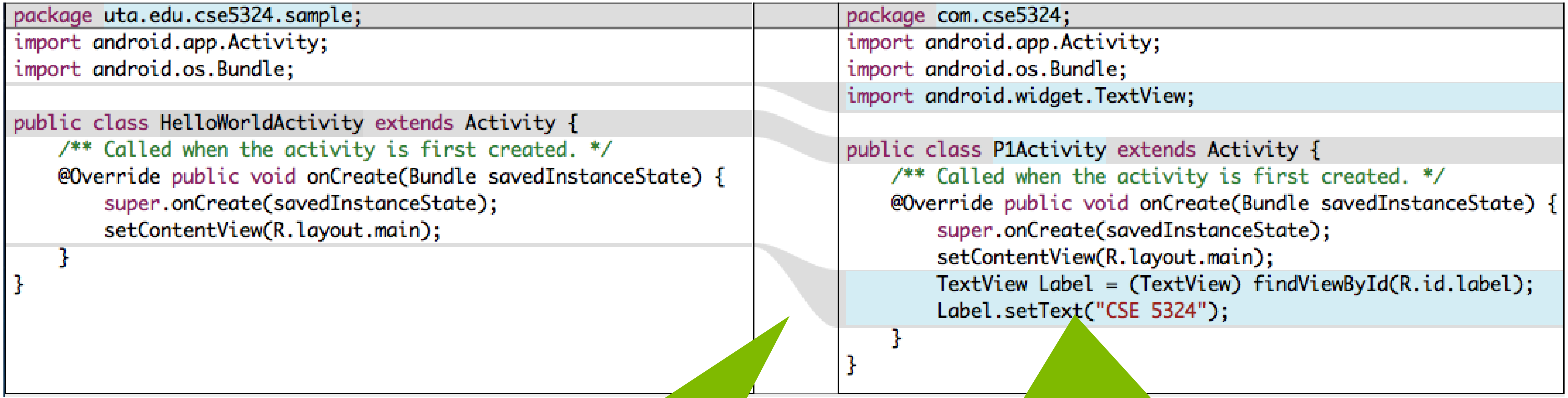
Correct Apps Per Subject

Higher percentage of TouchDevelop subjects finish 1,2,3 apps
Higher percentage of Android subjects finish 4 apps



Subject Reusing Android Apps Between Tasks

"Hello world" template (left) vs. "Print CSE 5324" (right)



```java
package uta.edu.cse5324.sample;
import android.app.Activity;
import android.os.Bundle;

public class HelloWorldActivity extends Activity {
    /** Called when the activity is first created. */
    @Override public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.main);
    }
}
```

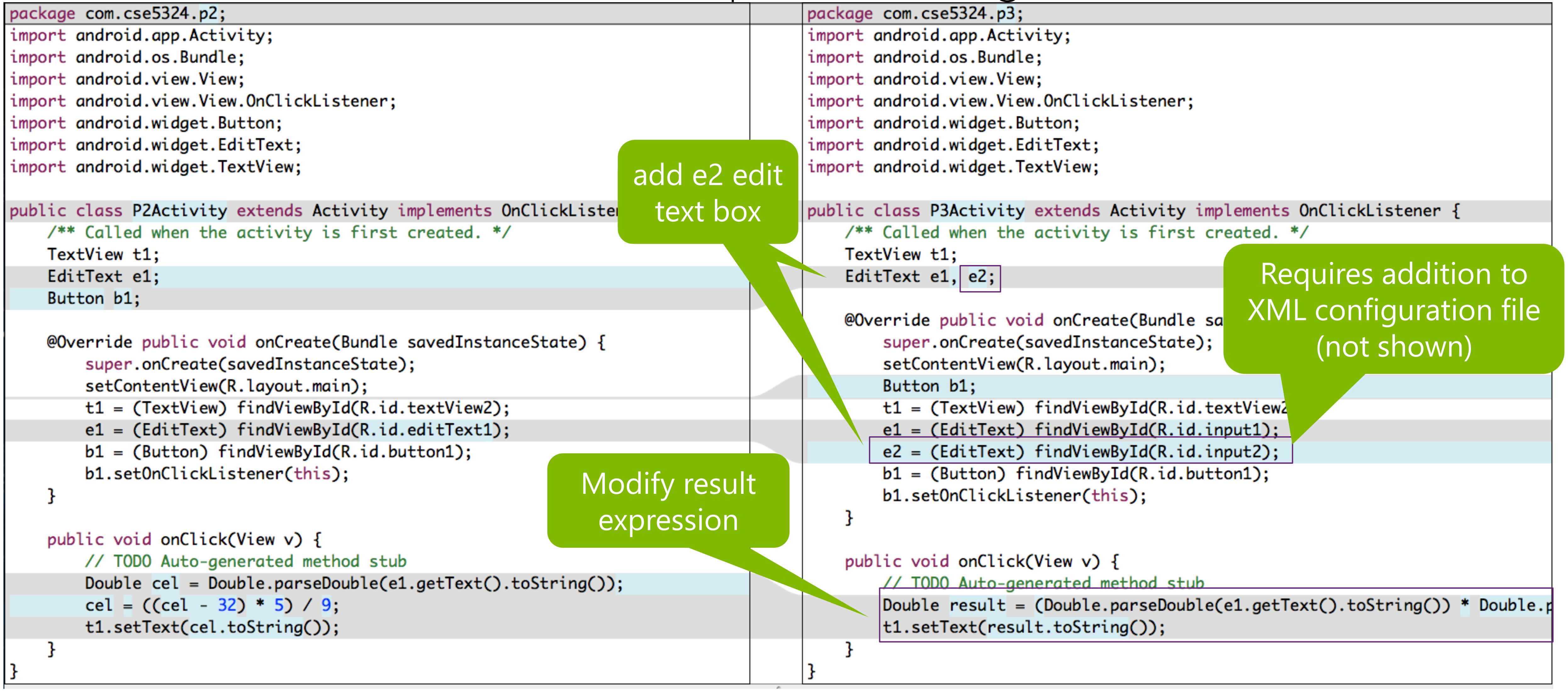
```java
package com.cse5324;
import android.app.Activity;
import android.os.Bundle;
import android.widget.TextView;

public class P1Activity extends Activity {
    /** Called when the activity is first created. */
    @Override public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.main);
        TextView Label = (TextView) findViewById(R.id.label);
        Label.setText("CSE 5324");
    }
}
```

Lookup text view and change text to ``CSE 5324''

Could have implemented this change without changing code, just replace "Hello World" string in XML configuration file

Subject Reusing Android Apps Between Tasks

°F to °C conversion (left) vs. Tip Calculator (right)



```
package com.cse5324.p2;
import android.app.Activity;
import android.os.Bundle;
import android.view.View;
import android.view.View.OnClickListener;
import android.widget.Button;
import android.widget.EditText;
import android.widget.TextView;

public class P2Activity extends Activity implements OnClickListe
    /** Called when the activity is first created. */
    TextView t1;
    EditText e1;
    Button b1;

    @Override public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.main);
        t1 = (TextView) findViewById(R.id.textView2);
        e1 = (EditText) findViewById(R.id.editText1);
        b1 = (Button) findViewById(R.id.button1);
        b1.setOnClickListener(this);
    }

    public void onClick(View v) {
        // TODO Auto-generated method stub
        Double cel = Double.parseDouble(e1.getText().toString());
        cel = ((cel - 32) * 5) / 9;
        t1.setText(cel.toString());
    }
}
```

```
package com.cse5324.p3;
import android.app.Activity;
import android.os.Bundle;
import android.view.View;
import android.view.View.OnClickListener;
import android.widget.Button;
import android.widget.EditText;
import android.widget.TextView;

public class P3Activity extends Activity implements OnClickListener {
    /** Called when the activity is first created. */
    TextView t1;
    EditText e1, e2;

    @Override public void onCreate(Bundle sa
        super.onCreate(savedInstanceState);
        setContentView(R.layout.main);
        Button b1;
        t1 = (TextView) findViewById(R.id.textView2
        e1 = (EditText) findViewById(R.id.input1);
        e2 = (EditText) findViewById(R.id.input2);
        b1 = (Button) findViewById(R.id.button1);
        b1.setOnClickListener(this);
    }

    public void onClick(View v) {
        // TODO Auto-generated method stub
        Double result = (Double.parseDouble(e1.getText().toString()) * Double.p
        t1.setText(result.toString());
    }
}
```

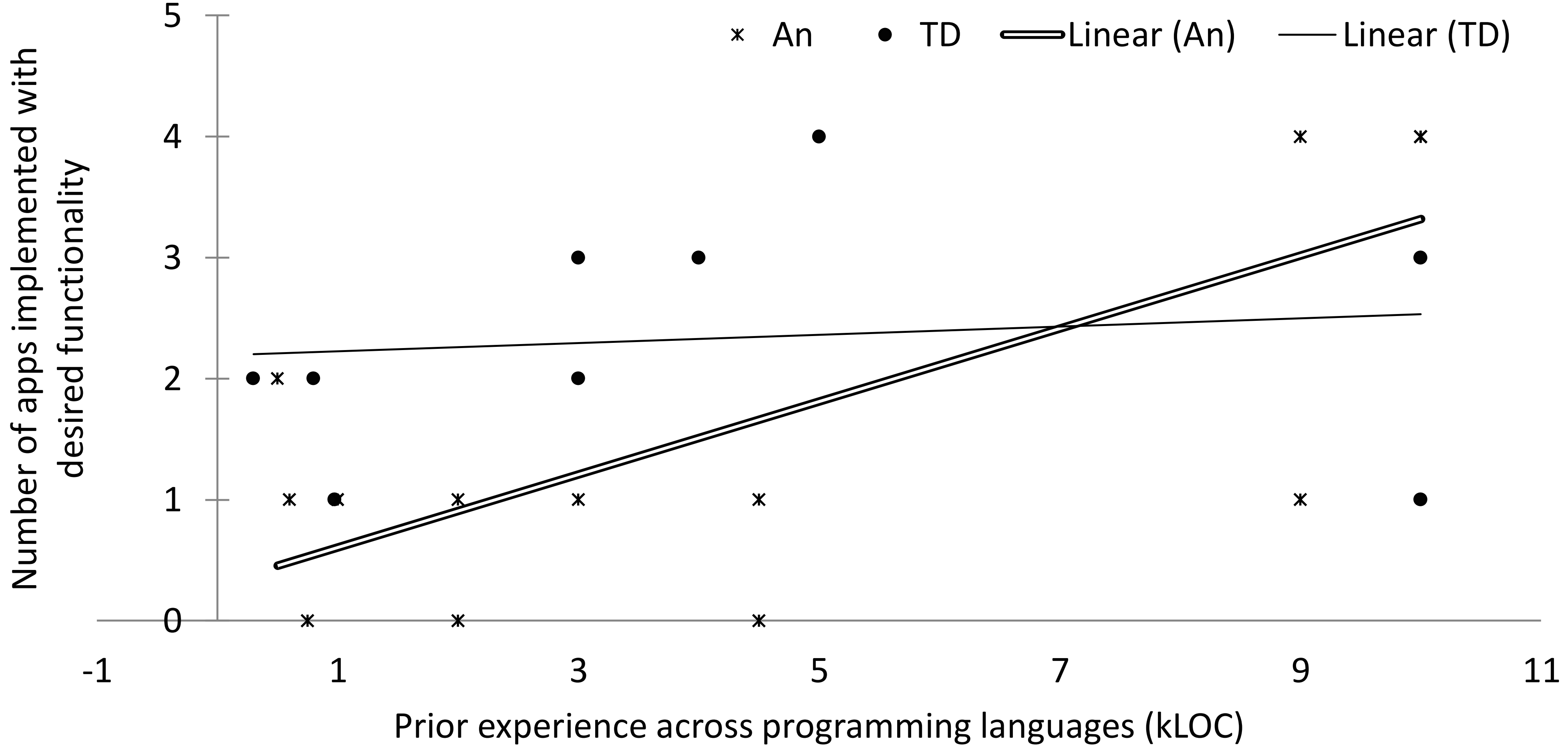
add e2 edit text box

Requires addition to XML configuration file (not shown)

Modify result expression

Prior Experience vs. Success in Experiment

Relation between app success and prior experience was much stronger for Android subjects



Post-Experiment Questionnaire

How many of the following have you done before this exercise?

Lines of code written, counting all programming languages: _________
(do not include plain html, but include JavaScript, C, C++, C#, Java, etc.)
Lines of (non-Android) Java code written: _________
Lines of C# code written: _________
Lines of Java for Android code written: _________
Lines of TouchDevelop code written: _________
Hours spent working with Eclipse (write Java code, etc..): _________
Hours spent learning TouchDevelop (watch video, read website, api, etc.): _________
Hours spent learning Android (watch video, read website, api, etc.): _________

Post-Experiment Questionnaire (2/3)

In completing this exercise, which problems did you encounter?

Preparing the IDE, emulator, etc.:

Developing particular apps:

Loading apps into the device:

Other (please elaborate):

In completing this exercise, which sources did you use (web sites, etc.)?

Samples that were part of the tool:

Official API documentation:

Examples found on the web:

Other (please elaborate):

Post-Experiment Questionnaire (3/3)

Comparing these sources with other documentation you have used in the past, how useful were the sources you used in this experiment?

Samples that were part of the tool:
Official API documentation:
Examples found on the web:
Other (please elaborate):

Which aspects of this exercise did you particularly enjoy?

Please let us know any additional comments you may have.

Sources Used By Subjects During Experiment

TouchDevelop subjects mainly used code samples
Android subjects: API sources and web sources

Source used	Android (%)	TouchDevelop (%)
Code samples	13	$\boxed{80}$
API	53	0
Web sources	53	20
Other sources	7	20

Acknowledgements and Credits

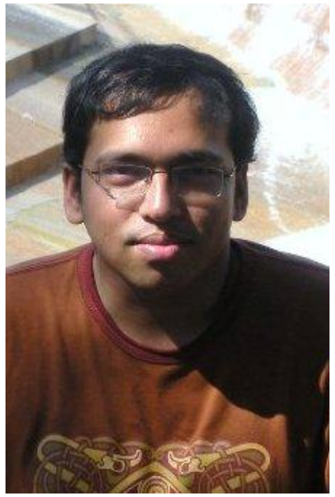
Microsoft Research Connections

For providing Windows Phones for the duration of the semester

National Science Foundation

This material is based upon work supported by the National Science Foundation under Grants No. 1017305 and 1117369. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

Tuan ("Tony") A. Nguyen
University of Texas Arlington



Sarker T.A. Rumee
University of Texas Arlington

Nikolai Tillmann
Microsoft Research Redmond