TouchGuru: Static Analysis for Mobile Development Environments

Peter Müller

ETH Zurich

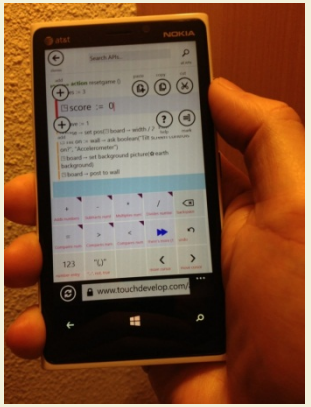
Joint work with

Lucas Brutschy and Pietro Ferrara

Challenges of Mobile App Development

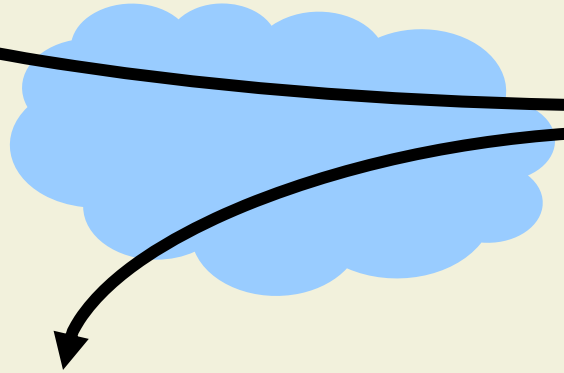
- Event-based execution
- Mobile environment
- Persistent storage
- Diverse platforms

- Education
- End-user programming
- Independent developers



Cloud-Based Static Analysis

```
action main ()
  var pic := senses → take camera picture
  pic → resize(100, 100)
end action
```



TouchGuru

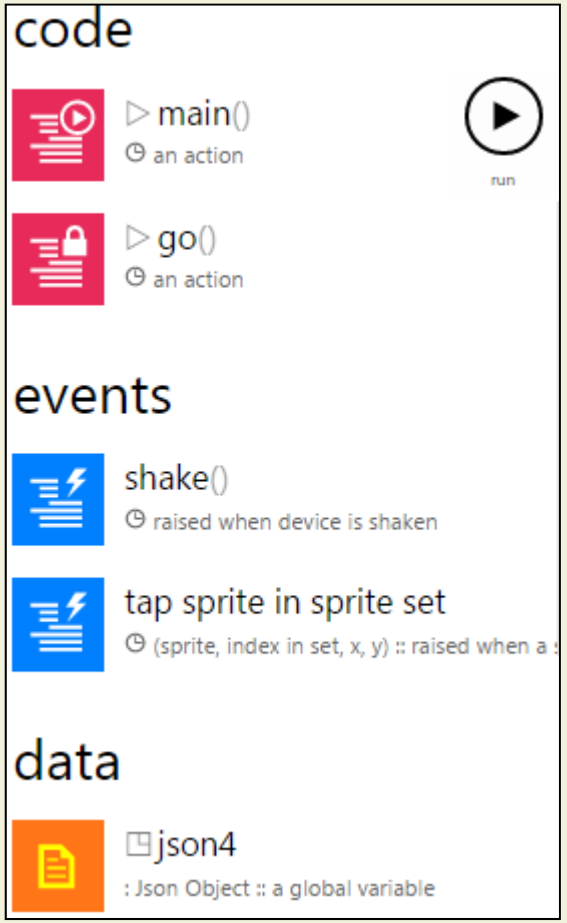
Analyze app for all:

- event schedules
- platforms
- environment interactions

```
action main ()
  var pic := senses → take camera picture
  pic → resize(100, 100)
  ✦ When calling resize: Object whose field/method is accessed might be invalid
end action
```

TouchDevelop Language

- Statically-typed, imperative
- Sequential execution
- Code is structured into
  - actions (procedures)
  - event handlers
- Data is stored in
  - local variables
  - persistent global variables
  - heap records
- Rich API



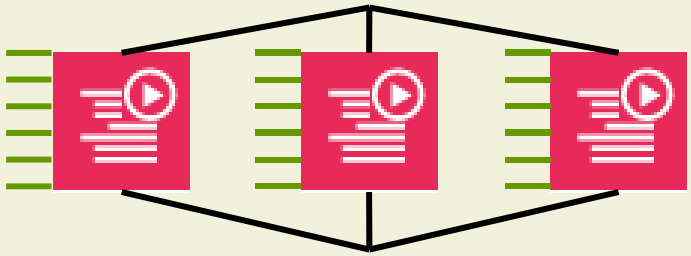
Overview of Static Analysis: Actions

- Semantics of statements

$$\mathbb{S} : Stmt \to (\Sigma \to (Label \to \Sigma))$$

- Semantics of any action

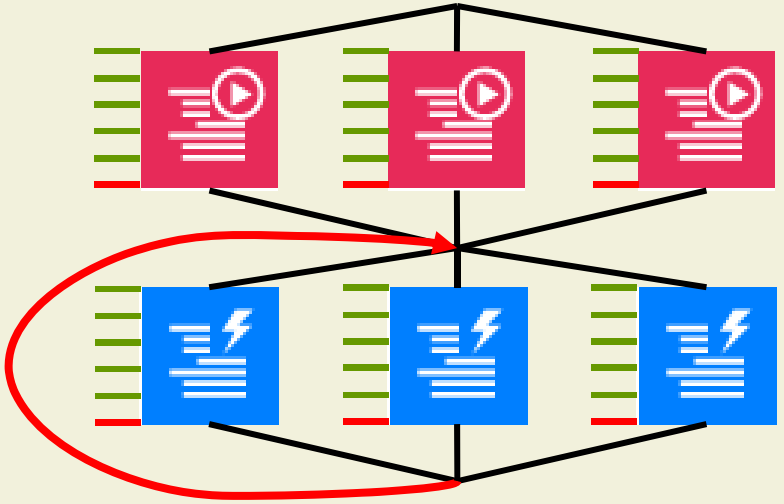
$$\mathbb{A}\,(\sigma) := \lambda l. \bigsqcup_{a \in Actions} (\mathbb{S}[a]\,(\sigma)\,(l))$$



Overview of Static Analysis: Events

- Semantics of any event

$$\mathbb{E}(\tau) := \lambda l. \bigsqcup_{e \in Events} \mathbb{S}[e] \left(\bigsqcup_{l' \in Exit} \tau(l') \right) (l)$$



- Semantics of an app

$$\mathbb{L}(\sigma) := lfp^{\sqsubseteq}_{\mathbb{A}(\sigma)} \lambda \tau. (\tau \nabla \mathbb{E}(\tau))$$

Challenge 1: Late Failing

- To increase robustness, many erroneous operations result in invalid value

```
if  0 ≤ index ≤ boards → size
then
  b := boards → at( index );
else
  b := media → create board;
b → post to wall;
```

- Abstract domain tracks origin of invalid-values for better error reporting

$$InvalidDom := Id \rightarrow \mathcal{P}\left(\{Valid\} \cup (\{Invalid\} \times Label)\right)$$

Challenge 2: Mobile Environment

```
var x := senses → acceleration stable → x
```

```
if senses → has accelerometer then
  var x := senses → acceleration stable → x
```

```
media → songs → random → play
```

```
if not media → songs → is empty then
  media → songs → random → play
```

```
if senses → has accelerometer then
  var x1 := senses → acceleration stable → x
  var x2 := senses → acceleration stable → x
  if x1 ≠ x2 then abort
```

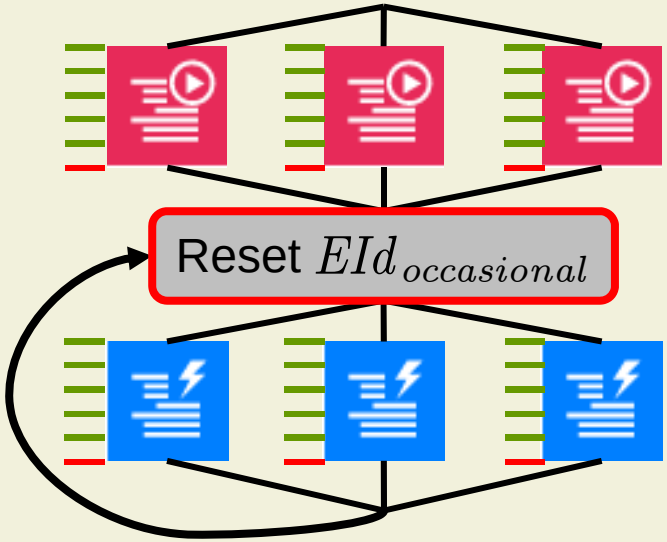
- Some aspects of the mobile environment
  - are stable
  - change occasionally
  - change frequently

Mobile Environment: Solution

- We model the environment via three sets of abstract identifiers

$$EId = EId_{stable} \cup EId_{occasional} \cup EId_{volatile}$$

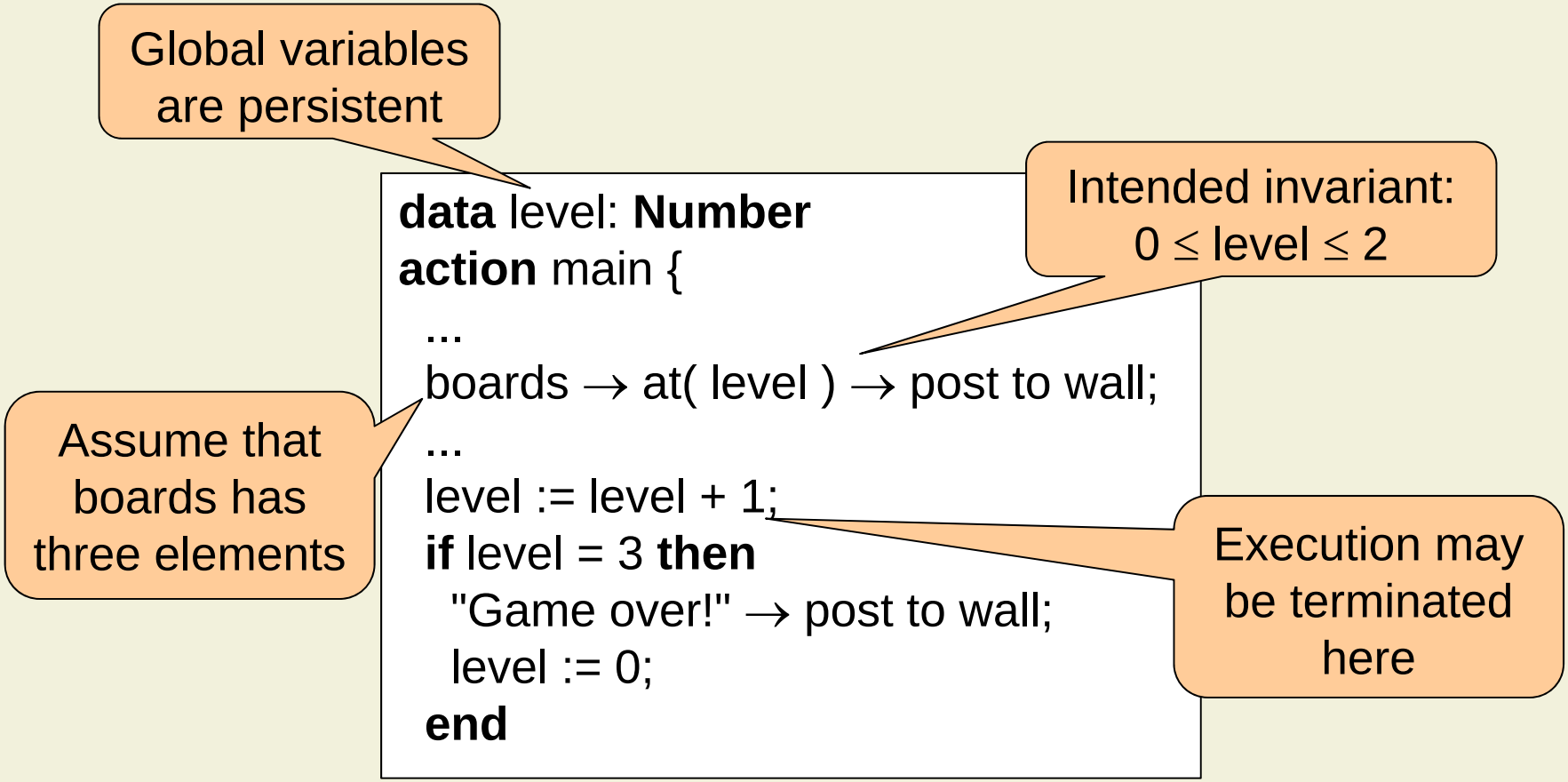
  - No information is tracked for “volatile” identifiers
  - Information for “occasional” identifiers is reset before each event handler



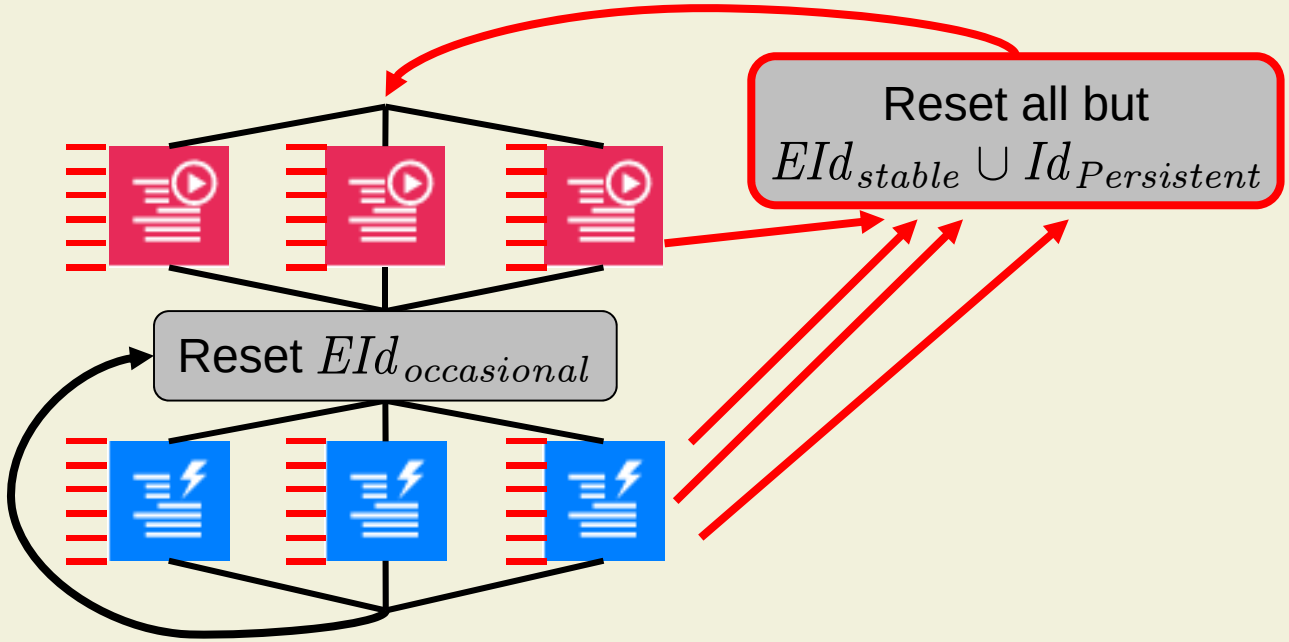
- Semantics of any event

$$\mathbb{E}(\tau) := \lambda l. \bigsqcup_{e \in Events} \mathbb{S}[e] \left(\bigsqcup_{l' \in Exit} \tau(l')|_{\overline{EId_{occasional}}} \right)(l)$$

Challenge 3: Persistent Storage



Persistent Storage: Solution



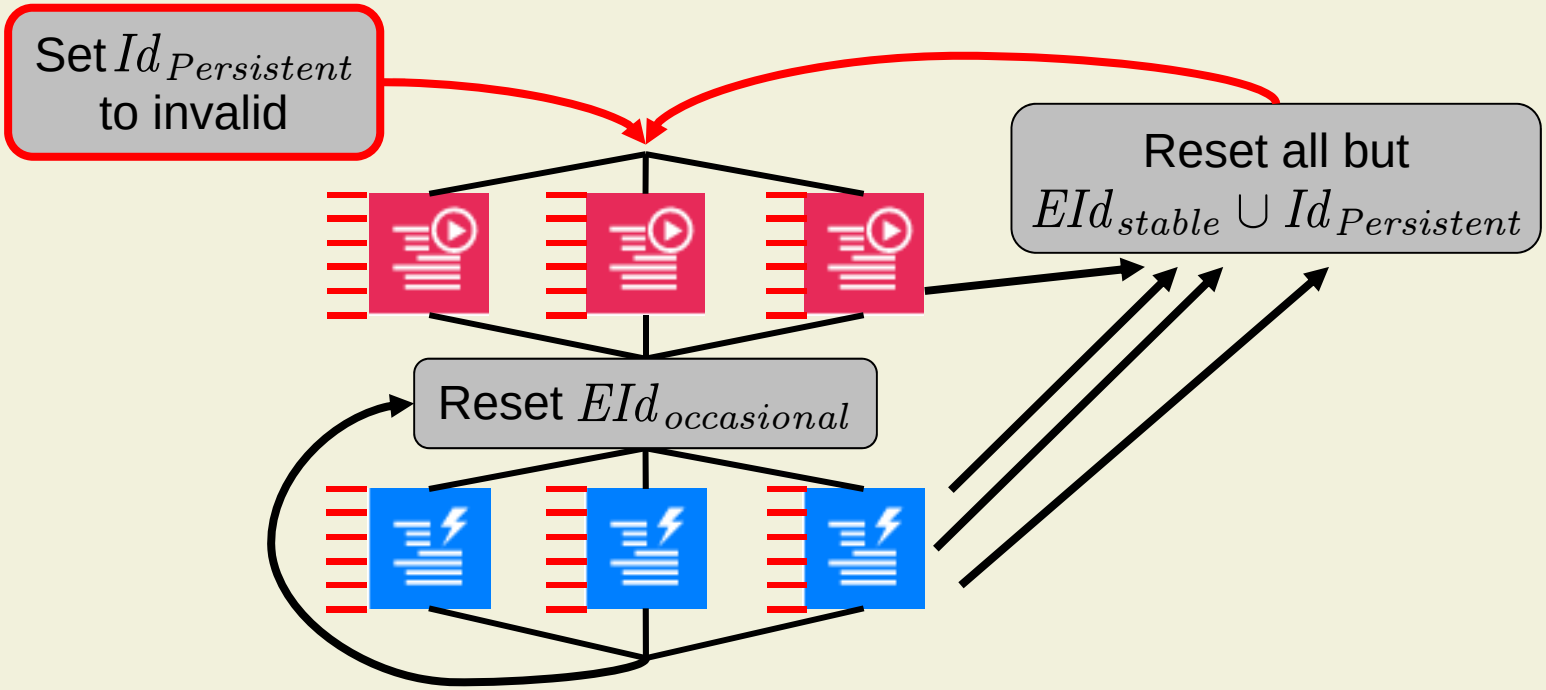
- Fresh execution from σ

$$\mathbb{L}(\sigma) := lfp^{\sqsubseteq}_{\mathbb{A}(\sigma)} \lambda\tau.(\tau \nabla \mathbb{E}(\tau))$$

- Next execution

$$\mathbb{R}(\tau) := \mathbb{L}\left(\bigsqcup_{l \in Label} \tau(l)|_{Id_{Persistent} \cup EId_{stable}}\right)$$

Persistent Storage: Solution (cont'd)



- Next execution
- Semantics of program

$$\mathbb{R}(\tau) := \mathbb{L}\left(\bigsqcup_{l \in Label} \tau(l)|_{Id_{Persistent} \cup EId_{stable}}\right)$$

$$\mathbb{P} := lfp^{\sqsubseteq}_{\tau_0} \lambda\tau.\,(\tau \nabla \mathbb{R}(\tau))$$

Experimental Results: Performance

- Implemented in Sample
  - Inter-procedural analysis
  - Value domain: octagons, strings, invalid
  - Heap abstraction: points-to and may/must container analysis

	#Scripts	#LOC		#Alarms		Time	
		Sum	Avg.	Sum	Avg.	Sum	Avg. [s]
root set	50,431	17,149,776	340.06	23,466	0.47	2d22h29'04"	5.03

- Complex analysis (three nested fixed points)

Experimental Results: Precision

- Manual inspection of 51 random scripts
  - 34 alarms
  - 85% precision: 29 true and 5 false alarms
- Ten most frequently executed scripts

Script	PID	#LOC	#Runs	Time [s]	#True	#False
Flip a Virtual Coin!	htmh	30	45,300	0.01	0	0
My Online Meetings	mpuj	333	41,100	1.47	0	3
WiFi/3G Swap	kmjn	44	40,800	0.07	0	0
Line Runner	dvvx	261	16,600	11.48	1	0
internet speedtest	qwzu	39	9,150	0.05	0	0
doodle jump	ajkc	221	9,000	9.14	1	0
where am I ??	kblp	98	8,100	0.40	2	0
CloudHopper	wbxsa	255	8,050	19.43	1	9
BreakIt! Touch	zids	180	7,700	3.98	0	2
doodle jump	ybcy	221	6,700	6.10	1	0
Total		1,682		52.13	6	14

Conclusions

- TouchGuru is an efficient and precise analysis for TouchDevelop
- Some of the addressed challenges occur also in other mobile platforms
- TouchDevelop is an attractive research platform
  - Simple language, small scripts
  - Cloud support