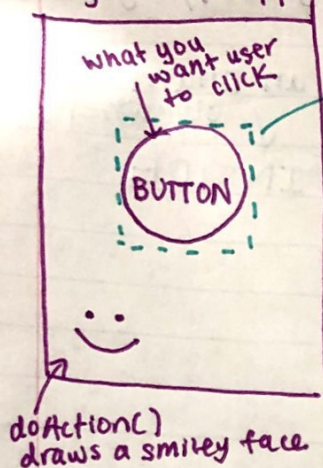


Topic: State machines with Essential Geometry

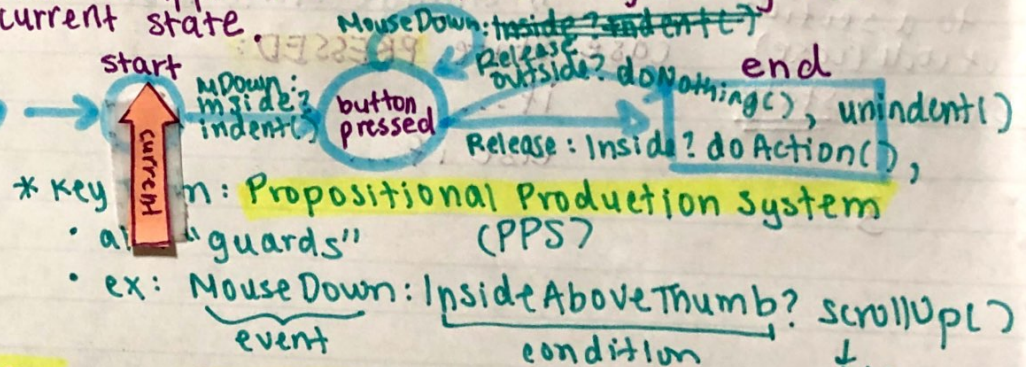
my Android App



① Background: what exactly are s. machines and essential geometry?

actual boundaries of your View object

a) **State machines**: basically a flowchart outlining what an application should do regarding the current state.



* Key: **Propositional Production System (PPS)**

• all "guards"

• ex: MouseDown: Inside Above Thumb? scrollUp()

b) **Essential Geometry**: says what part of your view is **actually clickable / actionable** (views are default rectangular)



- only want the user clicking w/in circle to change anything.
- give a name to each part of the view:
 - ex: Inside Button, Outside Button
- calculate user touch vs. circle, ex: distance formula from circle center to touch

- ② outline of steps:
- App starts, nothing happened yet → state = START
 - User presses down inside button circle → state = PRESSED, indent()
 - User drags finger inside circle
 - User drags out of circle
 - If user releases in circle, doAction() else: doNothing()
 - un-indent()

Code → in the View → onTouch

```
public boolean onTouch (Motion Event e) {  
    geometry = essentialGeometry (new Point (e.getX(), e.getY()));  
    this is given to you in your HW!
```

Switch (state) {

case state. START:

// check if geometry is equal to INSIDE
// change state if so

case state. PRESSED:

if ...
else if ...

an Enum defined elsewhere
↓

INSIDE



makes it EZ to jump to a section of code (case) based on state

3