/dev/world

Aug 31-Sep 1, 2015

Intimate Interactions on Apple Watch

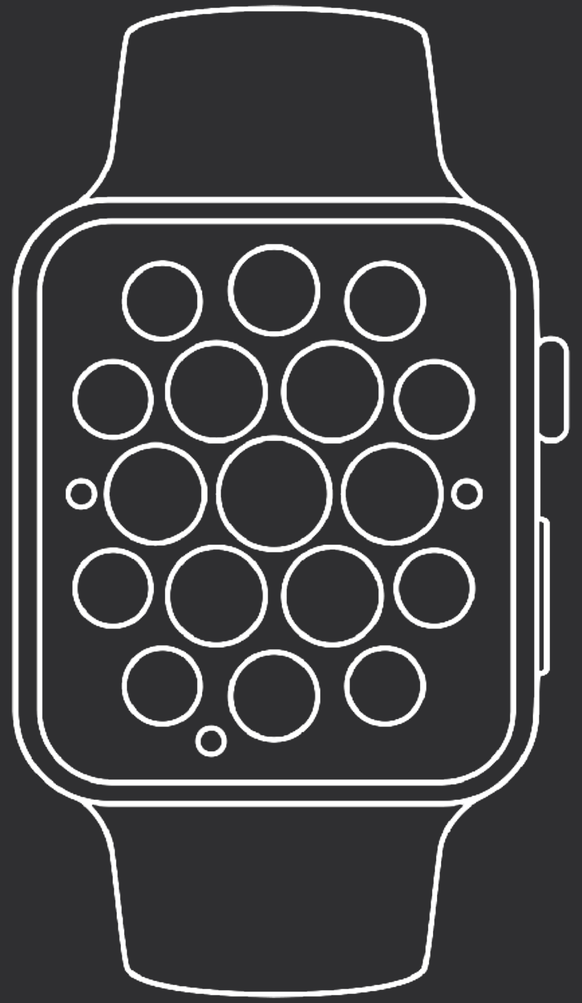
Phill Farrugia

Bilue

AUC

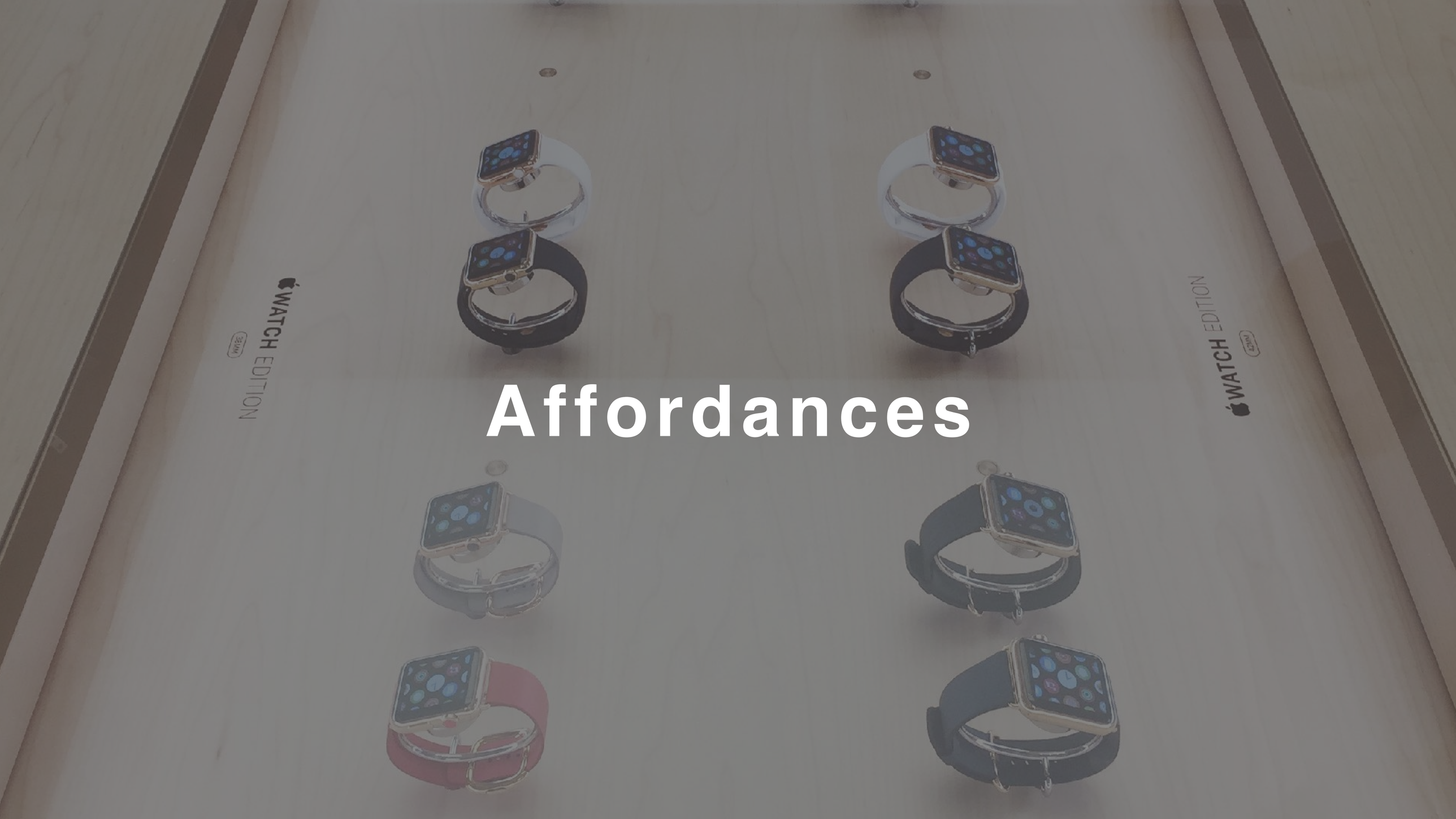
In Partnership With

RMIT UNIVERSITY



Overview

Affordances
Speed
Glances
Handoff
Notifications
Complications



Affordances

Smartphone usage

- In 2014, research in the UK showed smartphone owners reached for their phone up to 221 times a day between 7:30 am and 11:30 pm.
- Preferred their phone over desktop/laptop for smaller tasks.
- Different levels of interaction

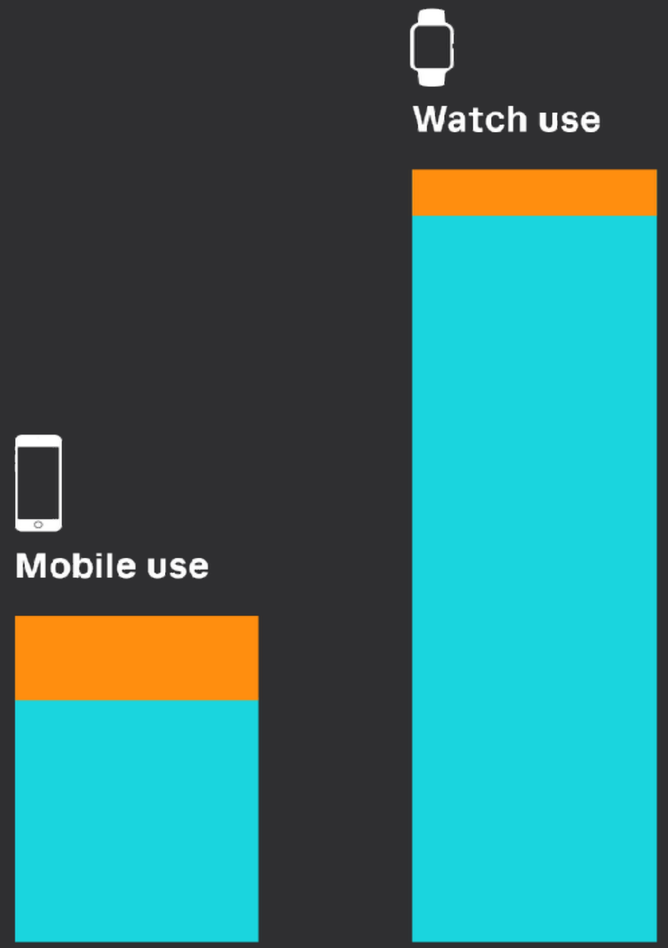
221 times/day

07:30

23:30

Micro-interactions

- Watches are designed for micro-interactions
- 100s of 1-5 minute engagements on Mobile each week
- 100+ 5 second engagements on Watch each week



Timely Information

- A watch is geared towards utility
- It becomes part of your daily routine
- Cuts down on amount of information we see in one take
- Get access to information quickly, easily

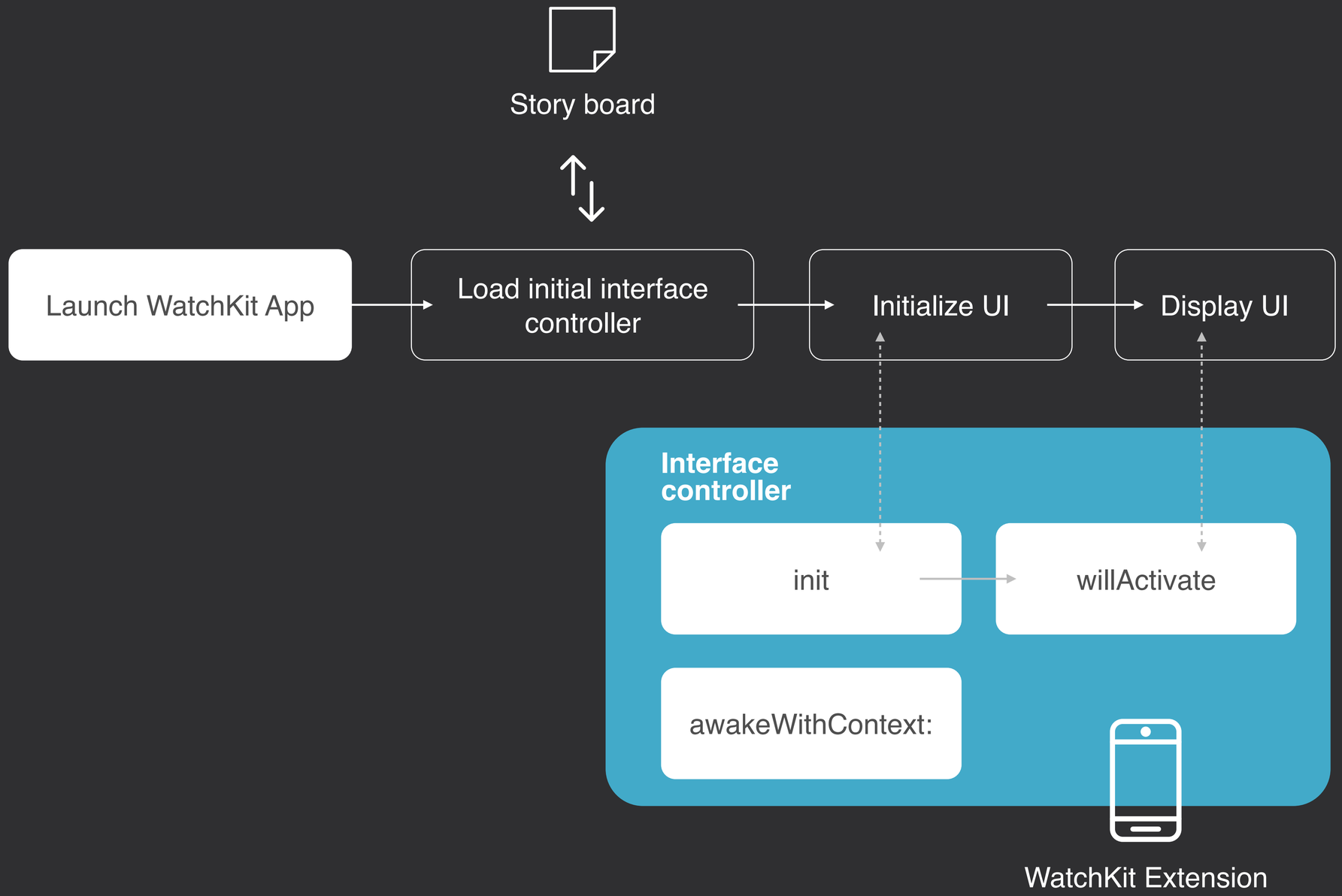
5x

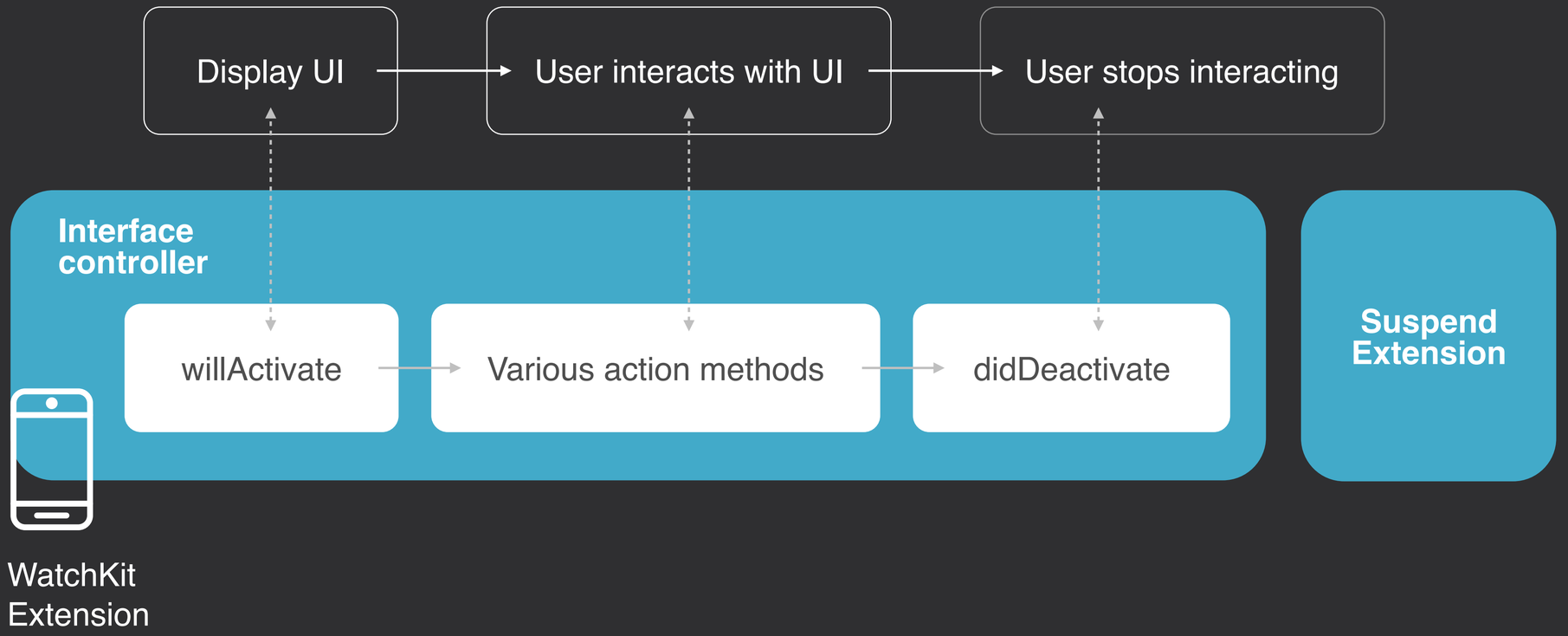
more information
than 30 years ago

Glance-able

- Present small sets of information
- Contextually aware (location, time, date, activity)
- Subtle, discrete and unobtrusive
- Distilled subset of entire iOS application
- Show only what is needed now, fast
- No complicated navigations, buttons, sliders, advertisements


Speed


Story board
Launch WatchKit App
Load initial interface controller
Initialize UI
Display UI
Interface controller
init
willActivate
awakeWithContext:
WatchKit Extension


Display UI
User interacts with UI
User stops interacting
Interface controller
willActivate
Various action methods
didDeactivate
Suspend Extension
WatchKit Extension

init

- Perform Interface ‘initalization’
- Anything that doesn’t involve data or calculation

awakeWithContext:

- Configure Interface based on contextual data
- i.e User clicks on table cell and ‘detail’ Interface is pushed with Model or ViewModel associated with the cell
- Update labels, image views, tables, trigger background network request.

willActivate

- Called every time interface is about to be visible
- Update information that has changed
- Do not ‘clear’ previous labels, images, table cells constantly
- Do not show custom spinner
- Do not continuously trigger new network requests. Use background tasks

```
- (void)performExpiringActivityWithReason:(NSString * _Nonnull)reason
usingBlock:(void (^ _Nonnull)(BOOL expired))block
```

willActivate

- Called every time interface is about to be visible
- Update information that has changed
- Do not ‘clear’ previous labels, images, table cells constantly
- Do not show custom spinner
- Do not continuously trigger new network requests. Use background tasks

didDeactivate

- Called every time interface is no longer visible (i.e. wrist down)
- Clean up any ongoing activity like timers, animations, calculations
- Your interface should not clear or reset labels, images, tables, elements
- Attempts to set labels, images, or table values are ignored

Wrist Down

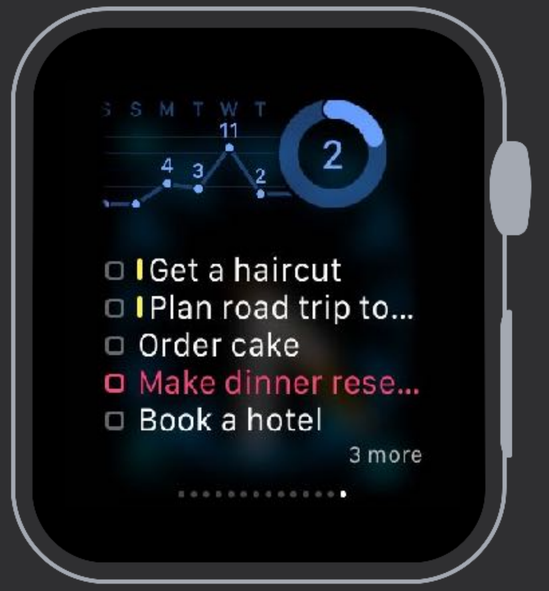
- Screen turns off
- viewDidDeactivate
- Save the state of your UI

Wrist Raise

- Screen turns on
- viewWillAppear
- Restore previous state of your UI

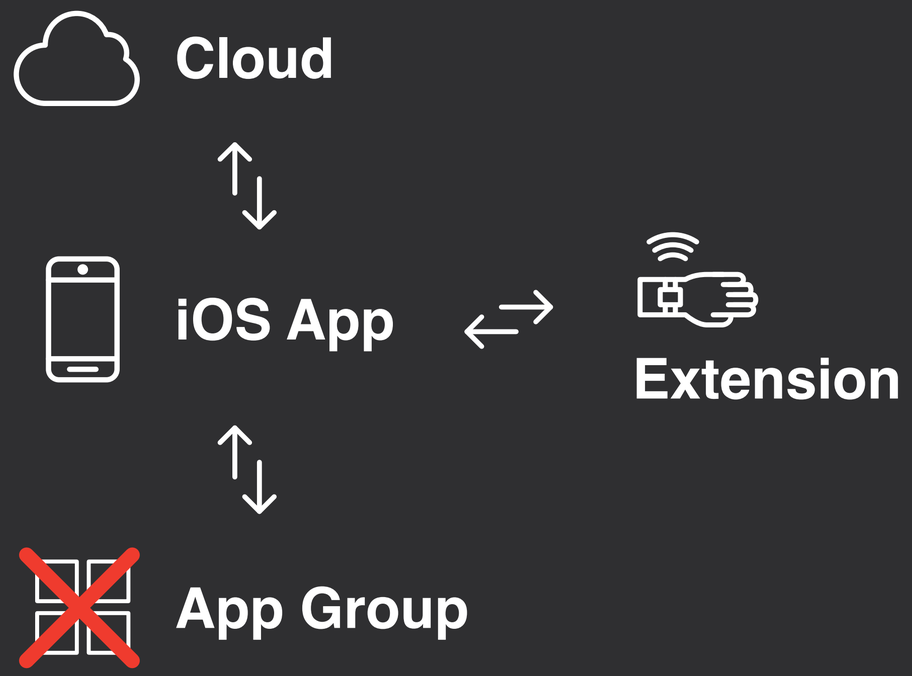
Glances

- Not all apps need a Glance
- Use context (time, location, date…) to be relevant
- Design for speed and simplicity not density



- Perform ALL setup before willActivate completes
- Avoid network requests at ALL costs
- Trigger network requests in background if needed

- Architect a solution to have the data ready and waiting if needed
- i.e. background location services updates iOS app and use Watch Connectivity Framework to transfer
- Push data changes from server to iOS app, use Watch Connectivity to transfer to watch

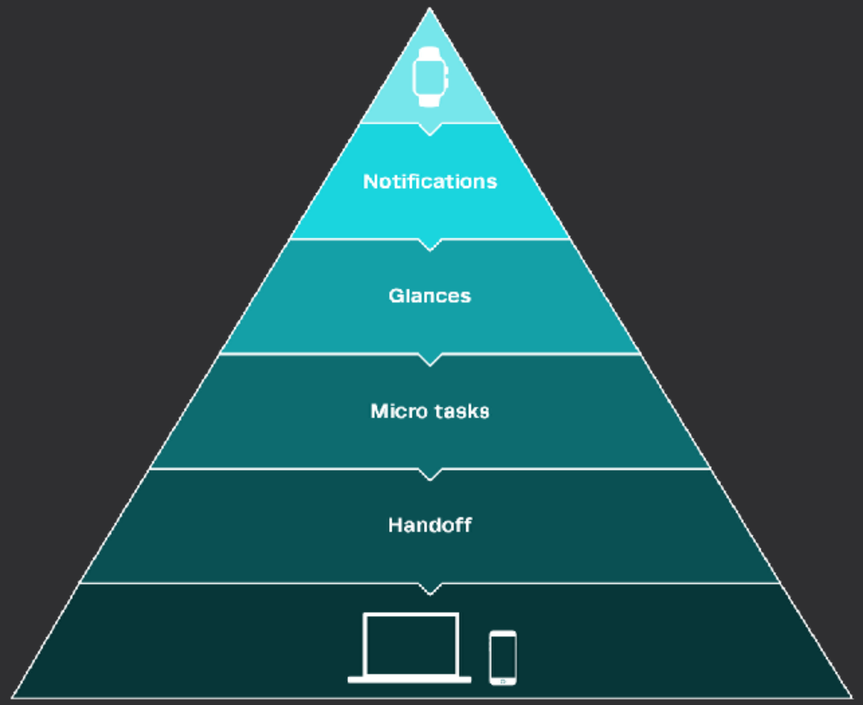


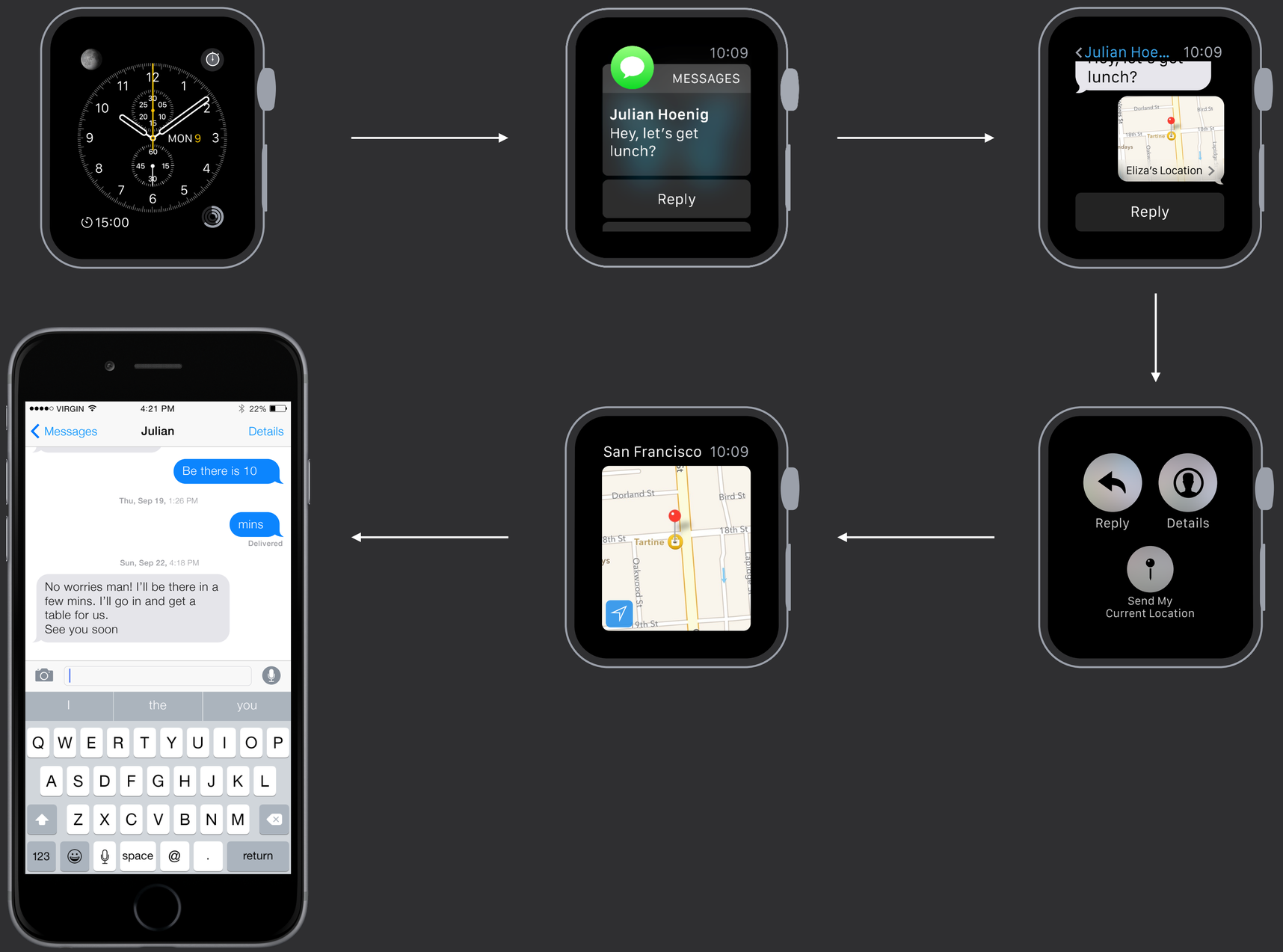
- Most regular entry point of your App
- Be instant
- Provide real utility
- Cut down on complexity
- Minimise network requests

- Focus on solving a single user need
- A ‘trustworthy’ Glance is reliable and seamless
- It becomes a consistent part of a user’s daily routine


Handoff

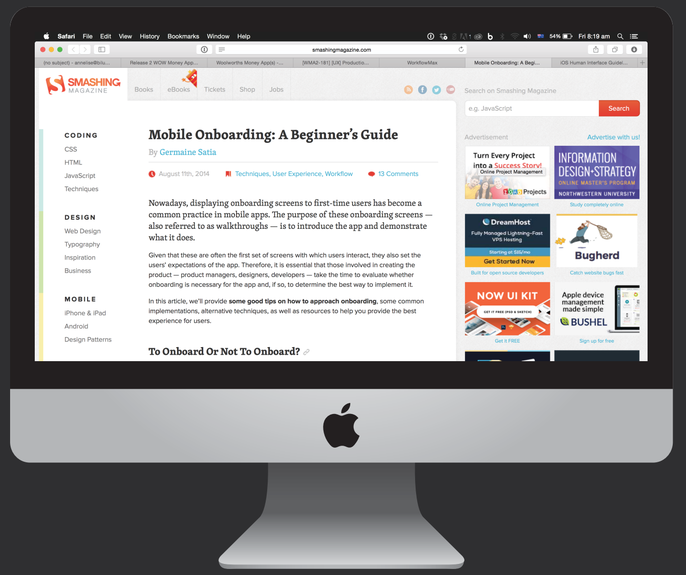
- Pyramid of engagement with your app
- Apple Watch, iPhone, iPad, Mac
- Seamless integration across devices
- Transitioning between different interaction patterns




MON 9
15:00
10:09
MESSAGES
Julian Hoenig
Hey, let's get lunch?
Reply
Julian Hoe...
10:09
lunch?
Eliza's Location
Reply
Reply
Details
Send My Current Location
San Francisco
10:09
Dorland St
Bird St
18th St
Tartine
8th St
Oakwood St
9th St
VIRGIN
4:21 PM
22%
Messages
Julian
Details
Be there is 10
Thu, Sep 19, 1:26 PM
mins
Delivered
Sun, Sep 22, 4:18 PM
No worries man! I'll be there in a few mins. I'll go in and get a table for us.
See you soon
I
the
you
Q W E R T Y U I O P
A S D F G H J K L
Z X C V B N M
123
space
@
.
return

updateUserActivity:UserInfo:WebPageURL:

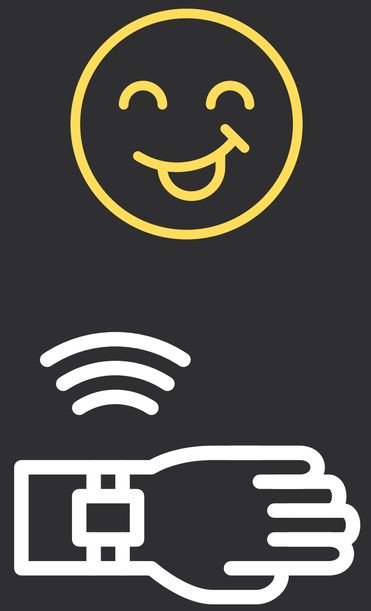
- Save significant user context within your app
- User taps on your Glance and opens relevant section of Watch App
- User is viewing an email in Watch App, unlocks their phone to type response
- Allows user to transition from micro-task (Glance) to fully fledged task (iOS Keyboard)



Notifications

- Sound
- Taptic Feedback
- Short look
- Long look
- Actions

- Deciding to action, or ignore notifications based on Taptic feedback
- Users interact with your app without even looking at a screen
- Non-visual interactions
- Subtle, fast, simple

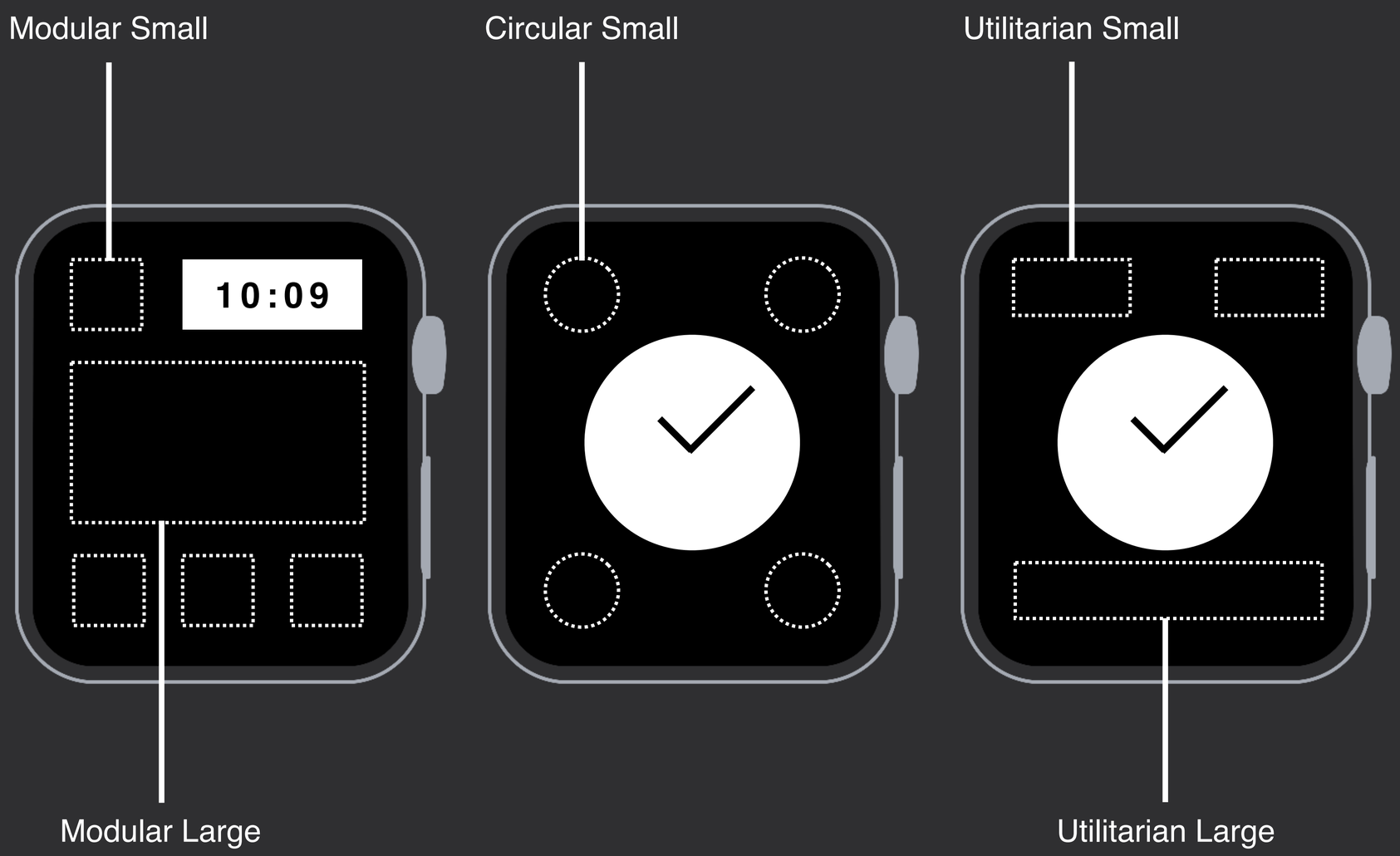


WKHapticType

```
enum WKHapticType : Int {
    case Notification
    case DirectionUp
    case DirectionDown
    case Success
    case Failure
    case Retry
    case Start
    case Stop
    case Click
}
```

Complications

- Small, modular windows into your app’s information
- Live on the watch face, alongside other apps
- Provide overview of information most valuable to wearer


Modular Small
10:09
Modular Large
Circular Small
Utilitarian Small
Utilitarian Large

- Data provided to ClockKit as entries in a timeline
- NSDate and Complication Template
- Complications rendered in background and data is cached
- User can travel back/forward in time