# Lesson 1

## Principles of Mobile Deign

* Respect user entered data
* Mobiles are personal
* Lives take precedence
* Work in all contexts
* User your sensors and your smarts
* User tasks usually take precedence
* Ensure consistency
* Respect information

Contextually aware: Ability to understand the network to which it is attached and pre-emptively gather information.

Composition: limitations and constraints, follow layouts and consistency.

Wrapper design

Layouts that wrap around all other components and content.

They organise info in a consistent template, info organised hierarchically, user can identify structure, increasing learnability with less errors.

# Lesson 2

## Scroll

Problem: Information on the page cannot fit in the viewport.  
Solution: Scrolls bars  
Anti-patterns:

* Allow users to become lost
* Tow dimensional scrolling
* Scrolling within scrolling
* Avoid drag and drop in scrollable area

## Notifications

Problem: Notify users without interfering with current process.  
Solution: Consistent method across OS, can be acted on or dismissed.  
Anti-pattern:

* Display serially (display all at one time using a method)
* Allow interrupt current activity
* Display to other device
* Determine your own method of display and interruption.

## Titles

Problem: Key elements need to be labelled  
Solution: Label content with titles, sub-titles with larger font. Consistency is key.  
Anti-patterns:

* Jargon and processes
* Repeating content

## Fixed menu

Design problem: Need to access options or controls across an app  
Solution: menu always visible, maybe docked to one edge. Useful for media players and cameras.  
Anti-patterns:

* Hiding fixed menus for viewport space.
* Stacking fixed menus.
* Adjacent fixed menus
* Accident activation with gestures.

## Lock Screen

Problem: Enter a lock/sleep state to prevent input, save power, prevent unauthorised access.  
Solution: displayed first when device is activated. Key info on how to unlock.  
Anti-patterns:

* Consuming power.
* Sending notifications to external devices.
* Creating your own lock screen
* Using key-combinations to unlock screens.

## Interstitial Screen

Problem: Delays before a request can be loaded preventing info from being displayed.  
Solution: Primarily a loading screen. Use when tech limitations prevent display. Use wait indicators.  
Anti-patterns:

* Using it to display ads
* Using it for every loading screen
* Locking during loading
* Modal (revealable menus).

# Lesson 3

## Context is Key

Users need access to info quickly and easily.  
Knowing when mobile interface is not good.  
Mapping: Standard control compatibility (volume slider goes up and down).  
Affordance: Object functions understood by their properties.  
Feedback: immediate, change of state, confirmations where user data can be lost.  
Constraints: reduce error, fit content, unimportant buttons are inactive but visible.

Position – Size – Shape – Contrast – Colour – Form

## Lists – Where would you use the following?

Infinite: need to display a vertical list but info is large and retrieval time is long.  
Thumbnail: a vert list with add info to assist understanding.  
Fisheye: a vert list where add info would prove valuable.  
Carousel: Need to present info where most are unique images.  
Grid: Need to present info where most are unique images.  
Infinite area: complex interactive visual info presented as a single image but is too large for viewport.  
Select: User needs to select 1 or more items.

## Control and Confirmation

Control: Respecting user’s data and input. Protecting against human error, data loss, unnecessary decision.  
Confirmation: necessary decision point met with actionable choice.

Don’t used confirmation because:

* They stop user’s goals from auto happening.
* Force users to read, understand and decide.
* Increase mental load.

Mobile are personal and therefore only need to authenticate once.

# Lesson 4

## UCD

Usability is the ease of access or ease of use of an application.

HCI is how humans interact with computers.  
UCD is from HCI and is focused on making apps meet user needs.

UCD places the user as the main focus of design. UX is how the user reacts to the design.  
this suits mobile dev because mobiles are personal.

## User Types

Information Overloader: Give you more info than is necessary. Keep on track and focused.  
Control Freak: Need it done their way or call the shots. Convince to work together, include them, let them make decisions.  
Devil’s Advocate: Don’t have ideas but don’t like your ideas. Give them roles or a task to fulfil.  
Dealing with Negativity: Focus on what’s wrong. Don’t take it personally, focus on product is wrong not you, list what is wrong, be optimistic and remain focused.

## Agile

Small steps towards building a solution.  
Work-out dev as it proceeds.  
Clients can change specs as product develops.

Why use it? Quick returns on investment. Generate momentum. Clients see development. Only design what is necessary.

## Y-Shaped Methodology

Focus on features, not platforms.  
Not restricted to mobile front end.  
Work on services and have ui use the services.