MAD

Mobile Application Design
Mobile is different

Smartphones and desktop computers are very different

Smartphones are actually more powerful than desktops in many ways
Mobile app design

A mobile app should do one thing and do it well

A mobile app should be as simple as possible, but no simpler

Further constraints
THINK “FAST FOOD” – SIMPLE, CHEAP AND ADDICTING.
Some key choices

- Market
- Device(s)
- Operating system
- Legacy components
- Deployment/distribution model
- Complete process
What is the target?

**Controlled set**
- Customers
- Employees

**Open set**
- Market
Devices (I)

> 1000 Android devices
Data collected during a 7-day period ending on May 7, 2019. Any versions with less than 0.1% distribution are not shown.

<table>
<thead>
<tr>
<th>Codename</th>
<th>API</th>
<th>Distribution</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gingerbread</td>
<td>10</td>
<td>0.3%</td>
</tr>
<tr>
<td>Ice Cream Sandwich</td>
<td>15</td>
<td>0.3%</td>
</tr>
<tr>
<td>Jelly Bean</td>
<td>16</td>
<td>1.2%</td>
</tr>
<tr>
<td></td>
<td>17</td>
<td>1.5%</td>
</tr>
<tr>
<td></td>
<td>18</td>
<td>0.5%</td>
</tr>
<tr>
<td>KitKat</td>
<td>19</td>
<td>6.9%</td>
</tr>
<tr>
<td>Lollipop</td>
<td>21</td>
<td>3.0%</td>
</tr>
<tr>
<td></td>
<td>22</td>
<td>11.5%</td>
</tr>
<tr>
<td>Marshmallow</td>
<td>23</td>
<td>16.9%</td>
</tr>
<tr>
<td>Nougat</td>
<td>24</td>
<td>11.4%</td>
</tr>
<tr>
<td></td>
<td>25</td>
<td>7.8%</td>
</tr>
<tr>
<td>Oreo</td>
<td>26</td>
<td>12.9%</td>
</tr>
<tr>
<td></td>
<td>27</td>
<td>15.4%</td>
</tr>
<tr>
<td>Pie</td>
<td>28</td>
<td>10.4%</td>
</tr>
</tbody>
</table>

https://developer.android.com/about/dashboards
Different operating systems
Legacy components

- Where do we came from?
- Where do we want to go?
- What assets do we want to keep?
- Which systems are we supposed to integrate?
Deployment model

- **Custom systems**
- **Closed world**
  - Centralized distribution
  - Dedicated sites
- **Market place**
Main development activities

THINK/PROTOTYPE  DESIGN  DEVELOP
Some key characteristics

- Mixed teams
- Sketches and prototyping
- Designers and not just computer scientists
How to start

• Mobile mindset
  • Focused, unique, charming, user-centered

• Different classes of users
  • Clearly identify your target(s): bored, busy, lost

• First impression is key
  • Limited/No help text
  • Characteristic and intriguing look and feel
  • Just a few seconds and the app...

http://www.netmagazine.com/features/10-principles-mobile-interface-design
Be bold

• Users are captured by unique design
• Users get tired of seeing the same old thing
• Do not use Android/Apple-supplied UI elements as a always-good solution
  • They are starting to look dated
Sketching
Something a bit more complete ...
Bucatini with Calabrian Style Pesto, with Fresh Ricotta and Basil-Infused Oil

DISCOVER THE RECIPE
How many features

- Users do not spend time discovering features
- Users do not complain about "advanced" features
- Users complain about features that do not work
Android design principles

- Enchant me
  - Delight me in surprising ways
  - Real objects are more fun than buttons and menus
  - Let me make it mine
- Simply my life
  - Keep it brief
  - Pictures are faster than words
  - Decide for me but let me have the final say
  - I should always know where I am
Further iOS design suggestions

• Use Layout to Communicate
• Avoid asking people to supply setup information
  • Focus on the needs of 80% of your users
• Launch in the device’s current orientation
• When your app restarts, restore its state so users can continue where they left off
• An iOS app never displays a Close or Quit option
  • Never quit an iOS app programmatically
Final suggestions

- Single and appropriate navigation model
- Minimal user inputs (through the proper means)
- Gestures are not really standardized
- Support orientations
- Communications
- Postpone sign up

http://www.netmagazine.com/features/10-principles-mobile-interface-design
Flat design

If your app looks outdated, users will note that skeuomorphic design
Flat design

- Not boring
- Ornamental elements are viewed as unnecessary clutter
- Bright, contrasting colors make illustrations and buttons pop from backgrounds
- Minimalistic nature

http://www.creativebloq.com/graphic-design/what-flat-design-3132112
Consistent layout

Can be very "expensive"

Extremely important

Design libraries exist to help decide which layout is the best for a particular problem
Anti-patterns

- Metaphor mismatch
- Idiot boxes
- Too many chart elements
- Oceans of buttons
Avoid PCisms

Images courtesy of Mobile Design Pattern Gallery by Theresa Neil
Development options
Model-View-Controller

**Model**
- Encapsulates application state
- Responds to state queries
- Exposes application functionality
- Notifies views of changes

**View**
- Renders the models
- Requests updates from models
- Sends user gestures to controller
- Allows controller to select view

**Controller**
- Defines application behavior
- Maps user actions to model updates
- Selects view for response
- Uses one for each functionality
Web-based solution
Real app
Webview-based solutions

Diagram:
- Your App
  - JavaScript
  - Bridge
- Platform
  - WebView
  - Canvas
  - Events
- Services
  - Location
  - Bluetooth
  - Audio
  - Sensors
  - Camera
  - etc.
Hybrid solution (PhoneGap)
Interpreted Solution (Titanium)

- A Titanium application is a JavaScript program
- Alloy framework
- Appcelerator Platform extends
  - Entire lifecycle of the application with debugging, testing, deployment, crash monitoring and analytic data collection
Solutions based on reactive views

Your App

JavaScript

Bridge

Platform

OEM Widgets

Canvas

Events

Services

Location

Bluetooth

Audio

Sensors

Camera

etc.
React Native

- Native mobile apps using JavaScript and React
- Uses the same fundamental UI building blocks as regular iOS and Android apps
- UI building blocks put together using JavaScript and React
- Pushed by Facebook
Flutter
Flutter is a mobile app SDK, complete with framework, engine, widgets, and tools.

Gives developers easy and productive way to build and deploy beautiful apps.

Also used for Fuchsia.

Dart (Flutter’s language) can be used to build web and server applications as well.

Learn Dart once, develop for five platforms.
Cross-compiled solutions
Cross-compiled solution (Xamarin)

- iOS C# UI
- Android C# UI
- Windows C# UI

Shared C# Backend
Native solution

[Diagram showing the integration of native code with OEM widgets and platform services such as Canvas, Events, Location, Bluetooth, Audio, Sensors, Camera, etc.]