



Web & Mobile & Designing User Experience

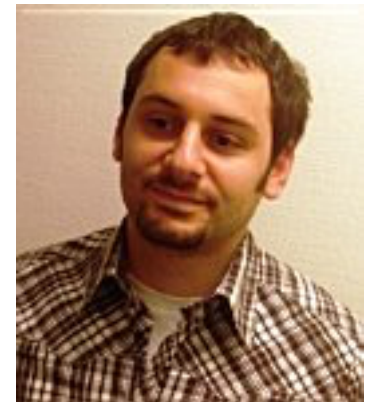
Summary and takeaways

Bahtijar Vogel, PhD in CS

“In the next century, planet earth will don an electronic skin.
It will use the Internet as a scaffold to support and transmit its sensations”

[Neil Gross 1999](#)

My research background and interests



Assistant Professor
2015-08-17

Bahtijar Vogel, PhD

bahtijar.vogel@mah.se



MALMÖ HÖGSKOLA

Background (Academia)

- 2015-present, **Assistant Professor @ MaH**, Web and Mobile Technologies, Open Architectures and Software Engineering
- 2014-2015, **Research Associate (Forskarassistent)** in Computer Science with specialization in Web and Mobile Technologies, and Software Architecture, Department of Computer Science, *Faculty of Technology*, Linnaeus University, Sweden.
- 2014, **PhD in Computer Science** “An Open Architecture Approach for the Design and Development of Web and Mobile Software” , Linnaeus University, Sweden.
- 2009-2012, **Phil. Lic.**, “Architectural Concepts: *Implications for the Design and Implementation of Web and Mobile Applications to Support Inquiry Learning*”. Licentiate Thesis, School of Computer Science, Physics and Mathematics, Linnaeus University, Sweden.
- 2009-2014, PhD student, “Mobile collaborative systems and service based web technologies”, Linnaeus University, Sweden.
- 2007, MSc, Computer Science, “*Cascading Web Services in Mobile Environments: Bridging Wireless and Wired Networks for Data Transactions* ” Växjö University, Växjö, Sweden.
- 2006, BSc, Computer Science, “BarSoft Management System: Pocket PC Application and its deployment”, South East European University, Tetovo, Macedonia.

Background (Industry)

- June 2013-November 2014, Energy Efficiency and Smart Homes Management System, Position: **Consultant as IT architect**, <http://interadria-ks.com>
- June 2013- November 2014, Human Recourses Software for the Government, financed by World Bank, Position: **Consultant as IT architect**, <http://interadria-ks.com>
- May 2006-July 2009, **Software Engineer**, <http://interadria-ks.com>
- March 2004-May 2006, **Software Developer**, <http://interadria-ks.com>

Two topics to be discussed

In a group of 5 - 6 please try to discuss and answer the following two topics based on the distributed materials from last week

- **Topic 1. Web vs native mobile app development:**
 - What are the benefits of HTML5 in mobile platforms?
 - What are the drawbacks of HTML5 in mobile platforms?
- **Topic 2. Choosing the right mobile platform/architecture and user experience design:**
 - What mobile platforms/architectures would you choose, and why?
 - How would you design the mobile user experience in order to make your app more usable?

How Mobile Devices are Revolutionizing User Interaction



Announcement of new Pope in St. Peters Square.
Photographs by Luca Bruno/AP, Michael Sohn/AP

<http://petapixel.com/2013/03/14/a-starry-sea-of-cameras-at-the-unveiling-of-pope-francis/>

Hwanyong Lee, Neil Trevett, Tom Olson, Victor Erukhimov, and Alon Ohbach. 2015. How Mobile Devices are Revolutionizing User Interaction. In *Proceedings of the 33rd Annual ACM Conference Extended Abstracts on Human Factors in Computing Systems*

Mobile Computing and Sensing Revolution

- Mobile is a major platform for computer human interaction innovation.
- The mobile industry has a huge market volume and lots of investment.
- Mobile devices have lowered in cost and while increasing in functionality.
- **A recent flagship smart phone has 21 sensors, including those for:**
 - *ambient light, RGB light, proximity, IR gesture, two cameras, IR autofocus laser, three microphones, touch, position (via GPS, Wifi, Cellular, NFC, and Bluetooth), accelerometer, magnetometer, gyroscope, pressure, temperature, and humidity.*

Challenges for cross platform development

- **Major platforms support:** Most of the developers would like to release apps for major mobile platforms (iOS, Android) and provide a consistent user experience (UX) across the platforms.
- **In-depth knowledge:** Developing an app for separate mobile platforms require in-depth knowledge of them and their SDKs.
- All these increases:
 1. **The cost of development,**
 2. **Ease of updating, and**
 3. **Time to market.**

Mobile HTML5

- **Business decisions**
 - Cheaper
 - Easy to maintain
 - Inside companies (Bring your own device policy)
 - No app installation required (pros and cons)
 - No Device Management application required
 - A mobile device is not necessary to execute the application, can be also done on a desktop browser

Challenges in mobile HTML5 solutions

- Browser fragmentation
 - Different browser have different HTML5 support/interpretation
- Monetizing an application is easier within app stores
 - However app stores want their shares
- Some native functionalities are not supported in HTML5
- Performance lacks behind native applications

Some decision factors

Decision criterion	Native approach	Mobile web approach	Cross platform approach
Quality of UX	Excellent	Very good	Not as good as native apps
Quality of apps	High	Medium	Medium to low
Potential users	Limited to a particular mobile platform	Maximum including smartphones, tablets and other feature phones	Large - as it reaches to users of different platforms
App development cost	High	Low	Medium to low
Security of app	Excellent	Depends on browser security	Not good
Supportability	Complex	Simple	Medium to complex
Ease of updating	Complex	Simple	Medium to complex
Time-to-market	High	Medium	Short
App extension	Yes	Yes	Yes

Dalmasso, I., Datta, S. K., Bonnet, C., Nikaein, N., & Antipolis, S. (2013). Survey , Comparison and Evaluation of Cross Platform Mobile Application Development Tools, 323–328.

Requirements for cross-platform

- **Multiple mobile platform support:** The framework must support several mobile platforms. Support for Android and iOS are very essential since they have the largest share in the application markets.
- **Rich user interface:** Currently the cross platform tools can not provide rich user interface (UI) as native apps. Since the success of an application highly depends on user experience of the interface, rich UI development should be incorporated. Support for sophisticated graphics (2D, 3D), animation, multimedia are necessary.

Requirements for cross-platform (cont'd)

- **Back-end communication:** Mobile devices promote an "always-connected" model in which the users are sharing material in social networking sites, watching videos, communicating via live chat, gathering information 24X7. There smooth support for backend **communication protocols** and **data formats** are absolute mandatory.
- **Security:** Applications developed by cross platform tools are not highly secure. Proper research needs to be carried out to secure the tools and applications.

Requirements for cross-platform (cont'd)

- **Support for app-extensions:** It is required to install app extensions on top of existing applications like in-app purchase/billing capability.
- **Power consumption:** It is an important issue now-a-days with thousands of smartphones and tablets are being activated daily. The generated applications must be optimized for power.

Requirements for cross-platform (cont'd)

- **Accessing built-in features:** The tools must be able to access the built-in features of a smart device. Use of camera, sensors, geo-localization and more features helps to provide better user experience.
- **Open source:** It attracts more of application developers and the developer community can participate in bug-fixes and further development. It is to be noted that this is not a technical requirement.

Cross platform development approaches

- Web apps
 - Based on widespread Internet technologies such as HTML5 and JavaScript which are based on different libraries/frameworks
- Hybrid apps
 - Combine advantages of the native world (distribution via the stores) with web technologies
- Interpreted apps
 - Use platform-specific native UI elements to interact with the user whereas the application logic is captured in a platform-independent way
- Generated apps
 - Compiled just like a native app and a platform-specific version of the application is created for each target platform.

Xanthopoulos, S. & Xinogalos, S., 2013. A Comparative Analysis of Cross-platform Development Approaches for Mobile Applications.

<http://heikobehrens.net/2010/10/11/cross-platform-app-development-for-iphone-android-co—a-comparison-i-presented-at-mobiletechcon-2010/>

Cross platform development approaches: comparison

	Web	Hybrid	Interpreted	Generated
Marketplace deployment	No	Yes, but not guaranteed*	Yes**	Yes**
Widespread technologies	Yes	Yes	Yes	No
Hardware and data access	Limited	Limited	Limited	Full access
User interface and look & feel	Simulated	Simulated	Native	Native
User-perceived performance	Low	Medium	Medium	High

**E.g. simple web clippings content aggregators or a collection of links, may be rejected by Apple.*

***Apps can be distributed without difficulty but the experience that the app provides must comply with the app store development guidelines.*

Xanthopoulos, S. & Xinogalos, S., 2013. A Comparative Analysis of Cross-platform Development Approaches for Mobile Applications.

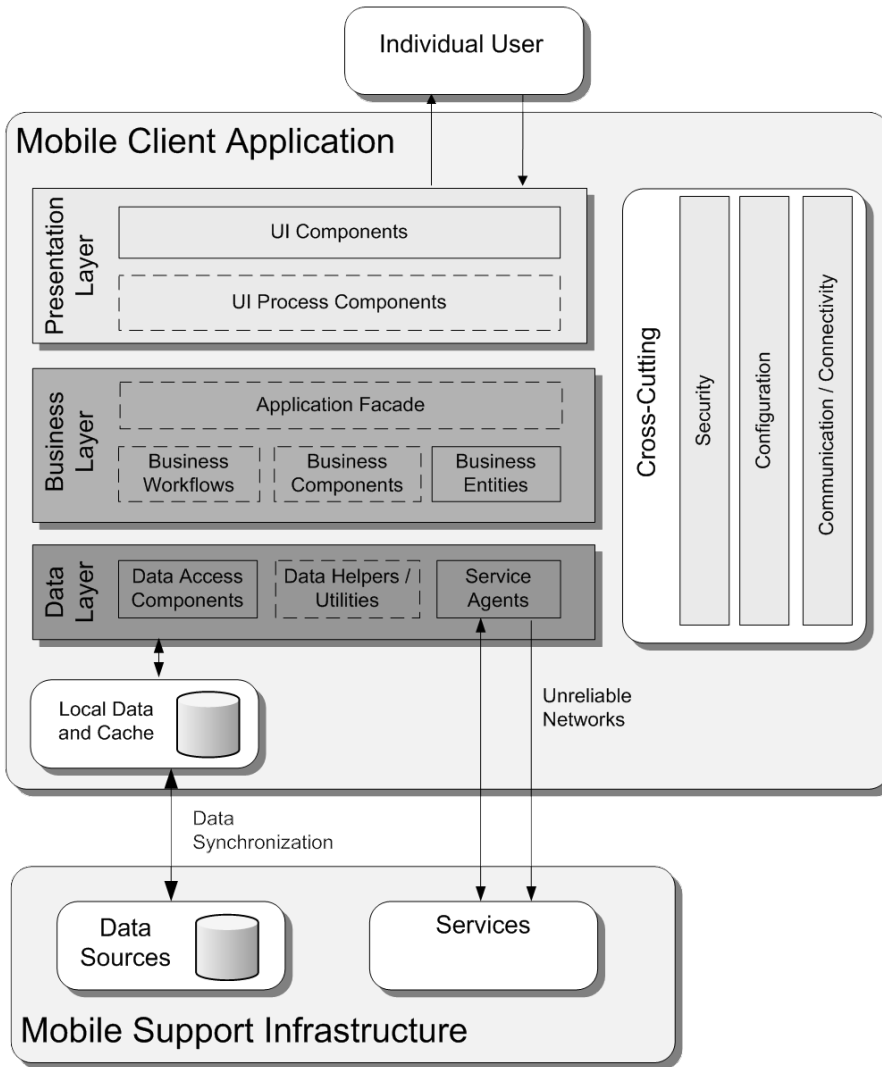
Software architecture and mobility: A roadmap

- Modern software-intensive systems are characterized not only by the movement of data, as has been the case in traditional distributed systems, but also by the movement of users, devices, and code.
 - systems that are deployed on novel computing platforms that have emerged during the past decade – smart phones, mobile robots, motes, sensors, and so on;
 - novel usage scenarios frequently demanding computation anytime, anywhere – applied to different domains (such as e-health, education, robotics)
 - amplified software engineering challenges distribution, decentralization, heterogeneity, resource constraints, context-awareness, real-time requirements, and so on.

A set of abstractions: mobile systems paradigms

- **Resource**, which is the data to be processed when delivering a system service,
- **Know-how**, which is the code component (i.e., algorithms) whose execution delivers the required service,
- **Computational component**, which is a processing thread,
- **Interaction**, which is any event involving two or more components, and
- **Site**, which is an execution environment that will host the data or run the code

Mobile Application Architecture Guide



Common mobile application architecture

Microsoft, 2008. Mobile Application Architecture Guide, Available at: http://robtiffany.com/wp-content/uploads/2012/08/Mobile_Architecture_Guide_v1.1.pdf.

Key Design Principles

- Key principles that will help you to create architecture that meets “best practices,” minimizes costs and maintenance requirements, and promotes usability and extendibility:

- Separation of concerns.** Break your application into distinct features that overlap in functionality as little as possible.

- Single Responsibility Principle.** Each component or a module should be responsible for only a specific feature or functionality.

- Principle of least knowledge.** A component or an object should not know about internal details of other components or objects.

Key Design Principles (cont'd)

- Don't Repeat Yourself (DRY).** There should be only one component providing a specific functionality; the functionality should not be duplicated in any other component.
- Avoid doing a big design upfront.** If your application requirements are unclear, or if there is a possibility of the design evolving over time, avoid making a large design effort prematurely.
- Prefer composition over inheritance.** Wherever possible, use composition over inheritance when reusing functionality because inheritance increases the dependency between parent and child classes, thereby limiting the reuse of child classes.

PhoneGap

- PhoneGap is a free and open source framework that allows you to create mobile apps using standardized web APIs for the platforms you care about.
- Standard web technologies such as HTML5, CSS3, and JavaScript
- Applications execute within wrappers targeted to each platform, and rely on standards-compliant API bindings to access each device's sensors, data, and network status.

PhoneGap

- **Adobe PhoneGap Enterprise:** Enterprise mobile application development and management across channels and platforms
- **The PhoneGap Developer App:** Develop locally then see the changes instantly on your mobile device with our cross-platform app
- **Package mobile apps in the cloud:** PhoneGap Build takes the pain out of compiling PhoneGap apps. Get app-store ready apps without the headache of maintaining native SDKs. Our PhoneGap Build service does the work for you by compiling in the cloud.
- **Weakness:** There is lack of support for native UI components, design patterns and dev tools. However, the developers are free to combine PhoneGap with another tool like JQuery or Sencha to produce a better UI for the apps.

PhoneGap: Supported features

	iPhone / iPhone 3G	iPhone 3GS and newer	Android	Blackberry OS 6.0+	Blackberry 10	Windows Phone 8	Ubuntu	Firefox OS
Accelerometer	✓	✓	✓	✓	✓	✓	✓	✓
Camera	✓	✓	✓	✓	✓	✓	✓	✓
Compass	X	✓	✓	X	✓	✓	✓	✓
Contacts	✓	✓	✓	✓	✓	✓	✓	✓
File	✓	✓	✓	✓	✓	✓	✓	X
Geolocation	✓	✓	✓	✓	✓	✓	✓	✓
Media	✓	✓	✓	X	✓	✓	✓	X
Network	✓	✓	✓	✓	✓	✓	✓	✓
Notification (Alert)	✓	✓	✓	✓	✓	✓	✓	✓
Notification (Sound)	✓	✓	✓	✓	✓	✓	✓	✓
Notification (Vibration)	✓	✓	✓	✓	✓	✓	✓	✓
Storage	✓	✓	✓	✓	✓	✓	✓	✓

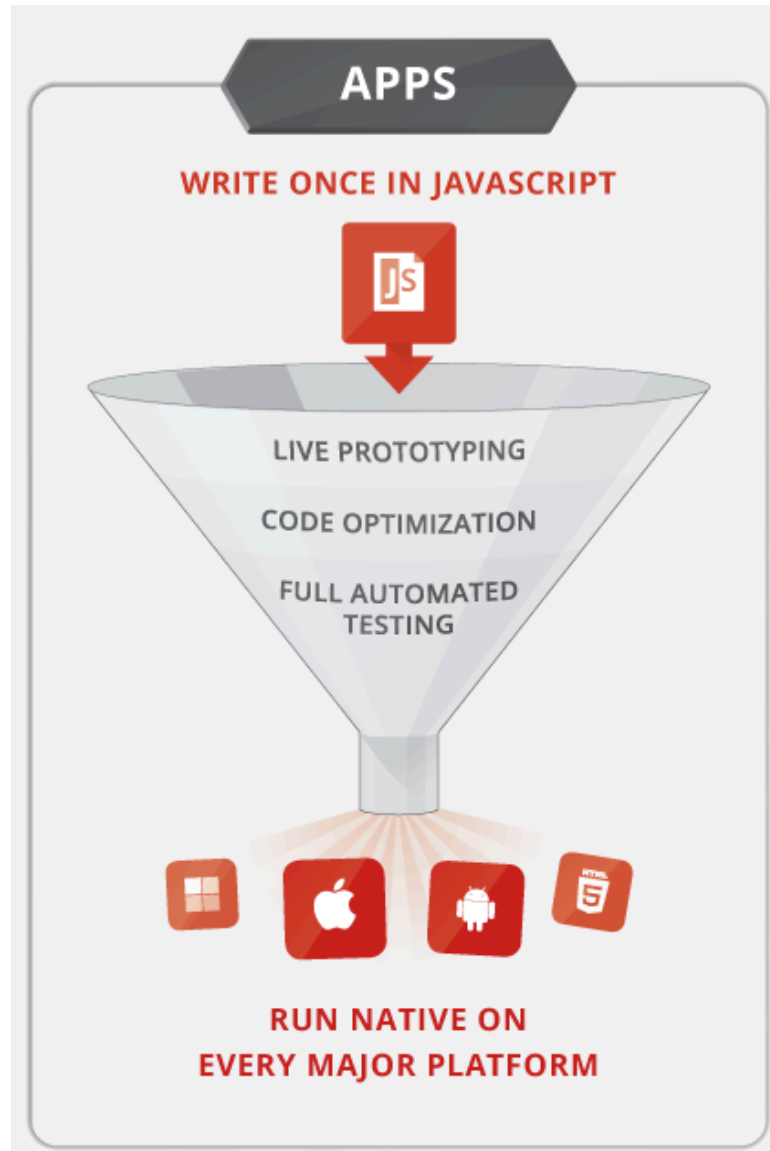
✓ - supported feature

X - unsupported feature due to hardware or software restrictions

The Appcelerator Platform

- Everything you need to create great, native mobile apps—all from a single JavaScript code base.
- **Build:** Write in JavaScript, run native on any device and OS
- **Test:** Verify apps with full mobile test automation
- **Connect:** Get mobile-optimized access to any data source
- **Measure:** See usage & adoption, detect crashes, tune performance
- **Weakness:** It has restrictive APIs and small set of phones are supported.

The Appcelerator Platform



Xamarin Platform

- **Build:** Write your apps entirely in C#, sharing the same code on iOS, Android, Windows, Mac and more
- **Test:** Find bugs before your users do, through cloud
- **Monitor:** Real-time crash and exceptions monitoring to improve your apps



Same-day support for new APIs

Xamarin is always up-to-date with the latest APIs from Apple and Google.



Connect with existing libraries

Includes more than 20,000 NuGet .NET libraries plus the Xamarin components.



Use your favorite IDE

Xamarin Studio on Mac or Windows, or Visual Studio.



Easily design iOS and Android apps

Build beautiful native user interfaces with our iOS and Android designers.

Xamarin Platform

- **Xamarin apps share code across all platforms:** Target iOS, Android, Windows and Mac with a single, shared C# codebase. Use the same language, APIs and data structures on every platform.
- **C# is the best language for mobile app development:** With Xamarin, you write your apps entirely in C#, sharing the same code on iOS, Android, Windows, Mac and more. Anything you can do in Objective-C, Swift or Java, you can do in C#.
- **Native UI, native API access & native performance.:** Xamarin apps are built with standard, native user interface controls. Apps not only look the way the end user expects, they behave that way too. This can't be achieved with other solutions.

Xamarin Platform

- **Some limitations:**
 - Since applications on Android require generating Java proxy types during the build process, it is not possible to generate all code at runtime.
 - These are the Xamarin.Android limitations compared to desktop mono.
 - Limitations for external libraries
 - Download time and the amount of storage space a Xamarin app uses on your devices

Other frameworks and libraries

- **AngularJS**
- **jQuery mobile**
- **Sencha Touch**
- **RhoMobile**
- **...**

Some developer advantages

- Easier and faster development process compared to building a native application for each specific OS.
- Reduction of required knowledge and skills since most cross platform frameworks uses easy to learn programming languages like HTML5 and JavaScript or one language. No certain knowledge about OS specific application programming interfaces (API:s) is required.
- The ability of having one codebase and be able to compile the source code to multiple operating systems makes development and maintenance a lot easier. This reduces both time and costs.
- Possible to expand sales of applications to more markets and app stores to increase the gain for developers, business model owners etc.

Designing your user experience

- Defining a functional scope for each screen
- Evaluating what data should be displayed
- Whiteboarding exercises to facilitate app design and identify discrete
- pieces of functionality
- Prototyping to get stakeholder and user feedback
- Using Agile to get software into the hands of beta testers

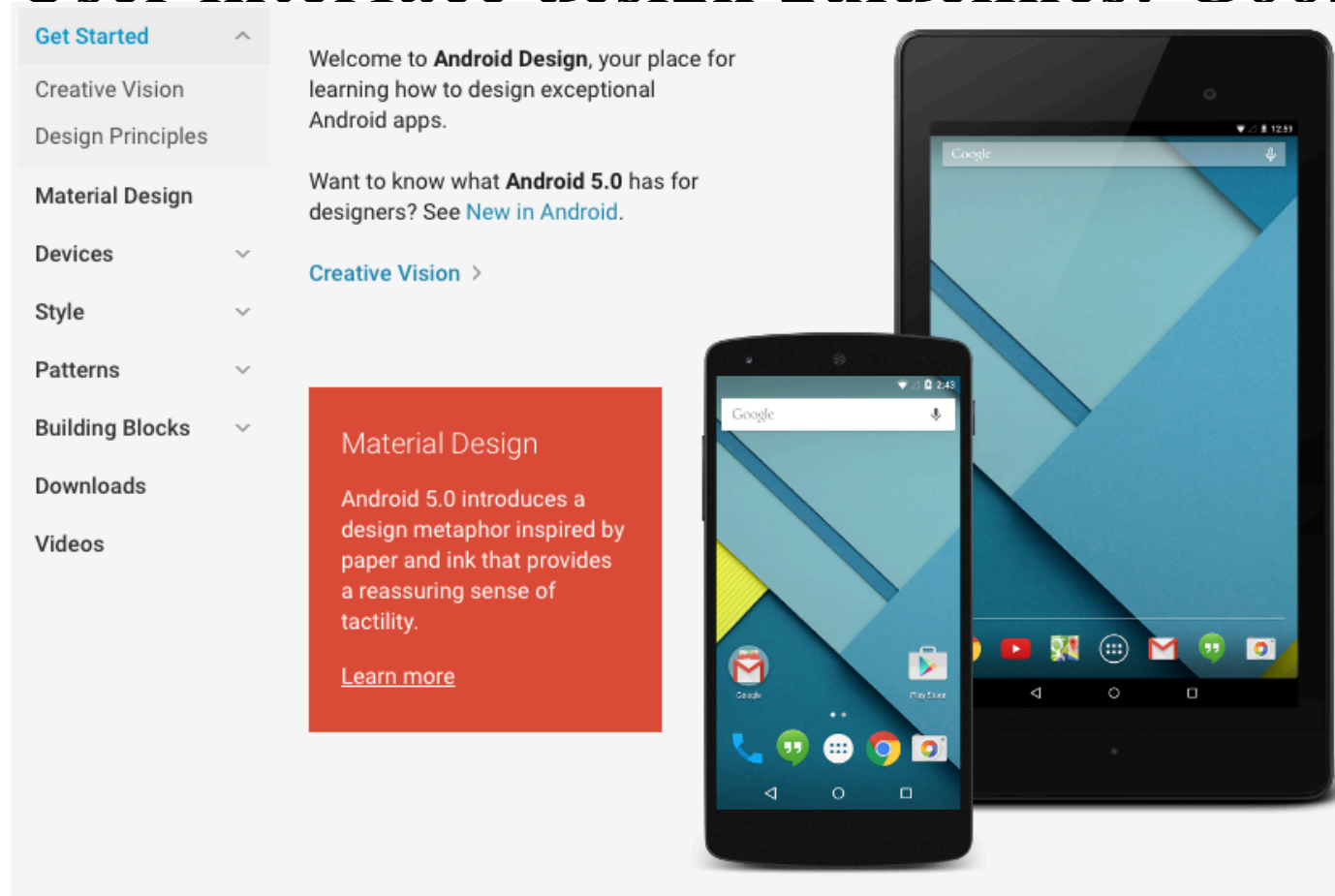
Making your application useable

- **Identifying the Scope of Each Screen** (limit the content and pieces of functionality contained in each screen)
- **Conforming to Platform Standards** (each mobile platform has its own set of best practices. Apple, Google, and Microsoft have all published and revised their user interface guidelines)
- **Separating Platform from Design** (when targeting an application for cross-platform deployment, you need to separate function from implementation. You can accomplish the same objective using entirely different user interface implementations)
- **Prototyping** (most development groups engage in some sort of prototyping exercises early in their development process. Prototyping is the process of developing less-than-complete implementations of software for the purpose of evaluating scope, functionality, and usability)

Making your application useable (cont'd)

- **Whiteboarding** (sketched mockup: the design can flesh out general business requirements)
- **Deciding What Data to Display** (A primary step in mobile application development is distilling what data must display on each screen (using simple diagraming paradigm)
- **Using Functional Prototypes** (before investing in a team of database developers and software testers, you need to create a functional prototype. The prototype can display canned data and doesn't need to include every screen)
- **Obtaining User Feedback** (one of the advantages of prototyping an application is that it provides a skeleton application that members of the targeted user base can test)
- **Using Agile Iterations** (application development teams no longer need to pigeonhole their efforts into following a waterfall-style development approach. Designing the user interface is a great time in the development cycle to embrace Agile development practices. Such practices center around not being locked down to requirements for long periods of time.)

User interface design guidelines: Google



<http://developer.android.com/design/index.html>

User interface design guidelines: Apple

iOS Human Interface Guidelines

iBooks

Search iOS Developer Library

On This Page

UI Design Basics

Designing for iOS

iOS App Anatomy

Adaptivity and Layout

Starting and Stopping

Navigation

Modal Contexts

Interactivity and Feedback

Animation

Branding

Color and Typography

Icons and Graphics

Terminology and Wording

Integrating with iOS

Design Strategies

iOS Technologies

UI Elements


Icon and Image Design

Revision History

Designing for iOS

iOS embodies the following themes:

- **Deference.** The UI helps people understand and interact with the content, but never competes with it.
- **Clarity.** Text is legible at every size, icons are precise and lucid, adornments are subtle and appropriate, and a sharpened focus on functionality motivates the design.
- **Depth.** Visual layers and realistic motion impart vitality and heighten people's delight and understanding.



<https://developer.apple.com/library/ios/documentation/UserExperience/Conceptual/MobileHIG/>

User interface design guidelines: Microsoft

Windows apps > Design

Design UWP apps

A great app starts with a great user interface. Learn how to design a Universal Windows Platform (UWP) app that looks fantastic on all Windows 10-based devices, from phones and tablets to PCs and Surface Hub.



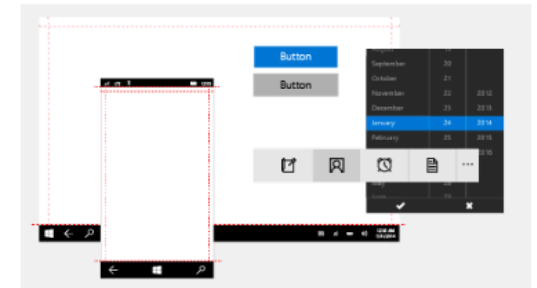
Design basics

Get started designing UWP apps: learn about the platform, UI design fundamentals, and responsive design techniques.



Guidelines

Browse the comprehensive list of guidelines for layout, controls, user interactions, text, and more.



Design downloads

Jump-start your project with design templates for PowerPoint and Adobe Illustrator.

<https://dev.windows.com/en-us/design>

Final remarks

- **“We expect revolutionary user interaction on mobile devices by standardization, research, and your imagination”**
- **The areas in which mobile developers need help to:**
 - Control, coordinate, and synchronize a diverse array of mobile sensors.
 - Write maintainable code for a heterogeneous mix of CPUs, GPUs, and DSPs.
 - Leverage dedicated vision hardware for minimized power.
 - Handle a diverse selection of emerging depth camera technologies.
 - Write code that is deployable across multiple devices, platforms and OSes.
 - Create fluid 60Hz experience on battery-powered mobile devices.
 - Employ best practices for UX design



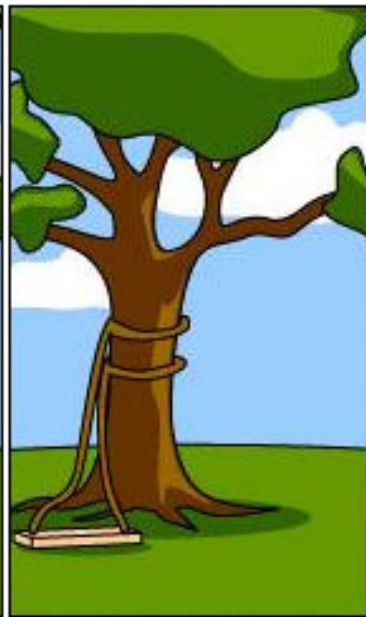
How the Customer explained it



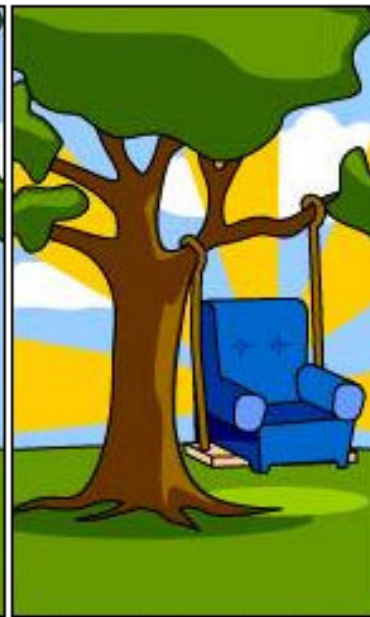
How the Project Leader understood it



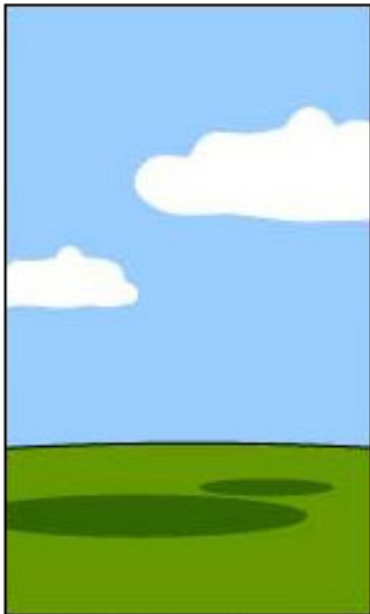
How the Analysts designed it



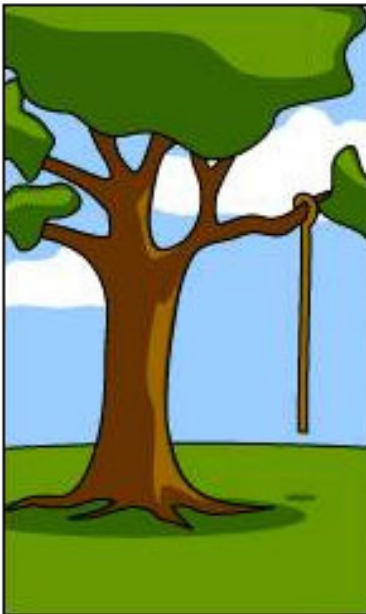
How the Programmer wrote it



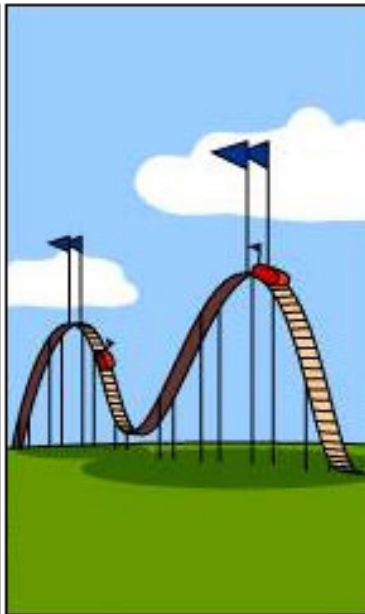
How the Sales Rep described it



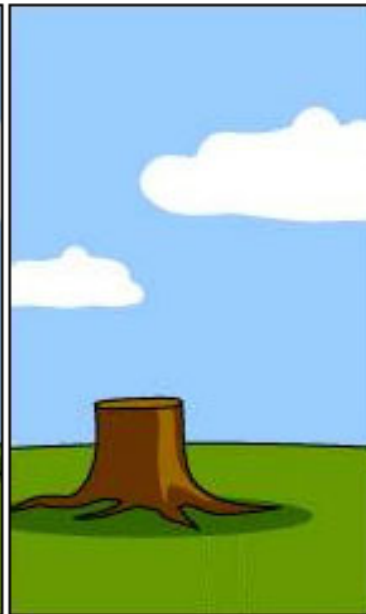
How the product was documented



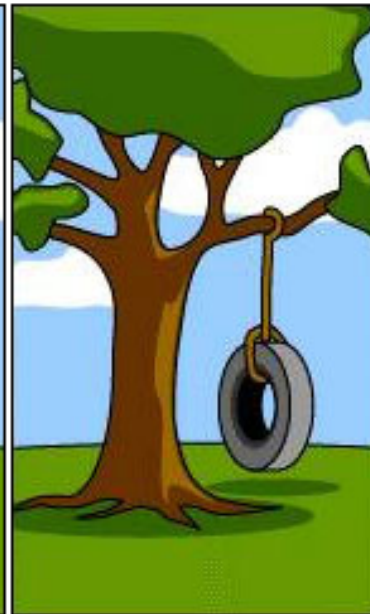
How the product was tested



How the customer was billed



How the product was supported



What the customer really needed