JQuery Mobile

1. Containers
   1. Web Application
      1. Cloud Provider
      2. Internet (HTTP/2)
      3. User Devices
   2. Cloud Backend
      1. Data Storage
         1. Data request
      2. Web Server
         1. HTTPs Request
         2. HTTPs response
      3. Load Balancer
         1. Https Request
         2. HTTPs response
      4. Cloud
   3. Virtual machines (CHEF)
      1. Hardware is abstracted, package up the OS, Package, and webserver code. Provisioned in minutes
   4. Containers (Docker)
      1. Hardware is abstracted and the OS are abstracted, package up packages, and webserver code
   5. Orchestration (Kubernetes)
      1. Compute becomes a fabric, you no longer manage the packing of applications
2. Mobile App Development
   1. Smartphone and smart Phone
      1. IBM SIMON
      2. Newton
   2. Different views on Applications
      1. Standalone app
         1. Runs on the smartphone (standalone app)
         2. Doesn’t need network / phone limits capabilities
      2. Distributed App
         1. Physically distributed
         2. Needs network / usage of external resources
   3. Distributed Apps
      1. Linked to backed systems?
      2. Distribution of tasks between phone and backend?
      3. Role of wireless network/telco?
   4. Mobile devices?
      1. Resource constrained
         1. CPU
         2. Memory
         3. Screen
         4. Network
            1. Intermitted connectivity
            2. No mouse
            3. No keyboard
            4. No …
         5. But
            1. New types of interaction

Touch-Screen

Shack/Move/Point

Voice/Speech

* + - * 1. Travels with user/trusted device

Rich context

* 1. Interface Design
     1. Less is more!!
     2. Some patterns (Neil)
        1. Springboard
        2. List
        3. Tabs
        4. Gallery
        5. …
  2. Application
     1. Requirements
        1. Purpose of the app (why do we need it)
        2. Functions / Interaction – less is often more
        3. Integration with other system/apps
        4. Integration with mobile device sensors
        5. Single versus Multiple Platforms
           1. Least common denominator?
        6. Software maintenance
        7. Testing
        8. Reliability, Availability, Security, privacy
        9. …
        10. Budget
     2. Some App Categories
        1. Online versus Offline
           1. Integration with backend systems
           2. Web services
           3. …
        2. Data or CPU driven
           1. Information / Entertainment

DB versus Eye candy

EB, local, remote

* + - 1. Native, Mobile web, hybrid
    1. Native Apps
       1. Program device specific
       2. Able to utilize resources better
       3. Better user experience
       4. Complexity? Time?
    2. Mobile Web
       1. Mobile web
          1. Focus on using browser as the environment
          2. Leverage web-technologies / web-standards
          3. Simpler development process
          4. Offline
          5. How to use device features
    3. Hybrid App
       1. Combine native & mobile web
       2. Build native app
       3. Use browser
       4. Approach
          1. Use native-code template
          2. Create browser object
          3. Deactivate browser security rules
          4. Link plugin libraries
          5. Push locally stored web-page into browser object
          6. This can be done on any mobile platform

🡪 cross platform approach

* + 1. Why HTML & JS
       1. New features of HTML5
       2. Consistent user experiences
       3. Less development costs
       4. Path to N-Screen design
    2. Key features of HTML5
       1. More semantic
       2. Cache, local storage, indexed DB, File API
       3. Geo-location & device access
       4. Web Sockets, Server sent events
    3. Frameworks
       1. Titanium
       2. Rhomobile
       3. Sencha
       4. Phonegap …
  1. JQuery Mobile
     1. JS lib for mobile devices
        1. Built on Jquery
        2. Large community
        3. Open and free
     2. Page Approach
        1. Internal Page
           1. Data-role = “page”
        2. External page 1
        3. External page 2
  2. Dialog
  3. Toolbars
  4. Footer
  5. Novel Aspects
     1. Scale
     2. Development speed, style
     3. Different delivery and access of applications
     4. Constrained browser, N-Tier server side
  6. Scale
     1. Web applications must scale
     2. Scalability key issue
        1. How to achieve this
     3. Management of state
        1. Client state, application state
     4. Challenge for underlying technology
        1. CGI <-> Threads <-> Web Farms <-> Web Garden
  7. Development speed and style
     1. Very short development cycles
     2. Heavy use of libraries
        1. Server and browser
     3. Less formal and more adaptive development
        1. You never know when the next tool is out
     4. With web 2.0 🡪 need to understand all aspects of application
        1. Generalist specialist
     5. Constantly searching for new techniques
        1. Comet
        2. Tricks / Techniques
        3. Connections to existing apps
  8. Delivery and Access
     1. Browser is used to access web functionality
        1. We can change frequently code
        2. No need for shipping discs
        3. Easy adding of new features
  9. Different Philosophies
     1. Minimalist approach
     2. Standard approach
     3. Cookies-cutter approaches
     4. Advanced approach
     5. Web 2.0