嵌入式计算机系统

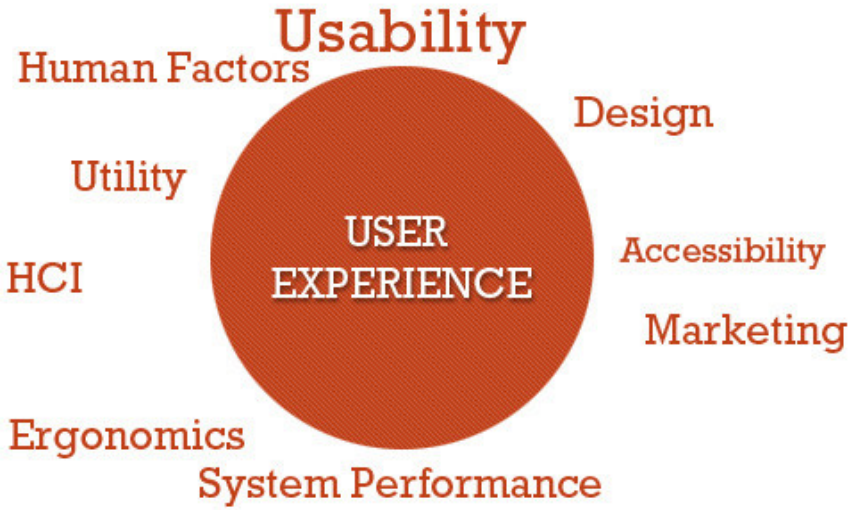
Lecture #9

MeeGo UX and MeeGo Graphics

内容来自于meego.com、MeeGo相关公开教程
以及“Task-centered user interface design” from http://hcibib.org/tcuid/

What is UX

- User experience (UX) is how a person feels when interfacing with a system.
- The system is generally denoted by some form of human-computer interaction(HCI)



Why UX is Important

- UX is core competency within today's software companies
- An expert in UX design is an indispensable part of software product team

A survey indicates that users prefer usability and good user experience over brand names alone

Why UX is Important

- User-centered design
  - It's not a perfect world, pervasive computing has not been realized.
  - We must prioritize and identify the areas that stand to gain the most from UX design and UX designers.

HCI design: Four basic activities

There are four basic activities in HCI Design:

1. Identifying needs and establishing requirements
2. Developing alternative designs
3. Building interactive versions of the designs
4. Evaluating designs

Some practical issues

- Who are the users?
- What are ‘needs’?
- Where do alternatives come from?
- How do you choose among alternatives?

Who are the users?

- Not as obvious as you think:
  - —those who interact directly with the product
  - —those who manage direct users
  - —those who receive output from the product
  - —those who make the purchasing decision
  - —those who use competitor’s products ???
- Three categories of user:
  - —primary: frequent hands-on
  - —secondary: occasional or via someone else;
  - —tertiary: affected by its introduction, or will influence its purchase.

Wider term: stakeholders

Who are the users?

- What are their capabilities? Humans vary in many dimensions!
- Some examples are:
  - —size of hands may affect the size and positioning of input buttons;
  - —motor abilities may affect the suitability of certain input and output devices;
  - —height if designing a physical kiosk;
  - —strength - a child’s toy requires little strength to operate, but greater strength to change batteries

What are ‘needs’?

- Users rarely know what is possible
- Users can’t tell you what they ‘need’ to help them achieve their goals
- Instead, look at existing tasks:
  - —their context
  - —what information do they require?
  - —who collaborates to achieve the task?
  - —why is the task achieved the way it is?
- Envisioned tasks:
  - can be rooted in existing behaviour
  - can be described as future scenarios

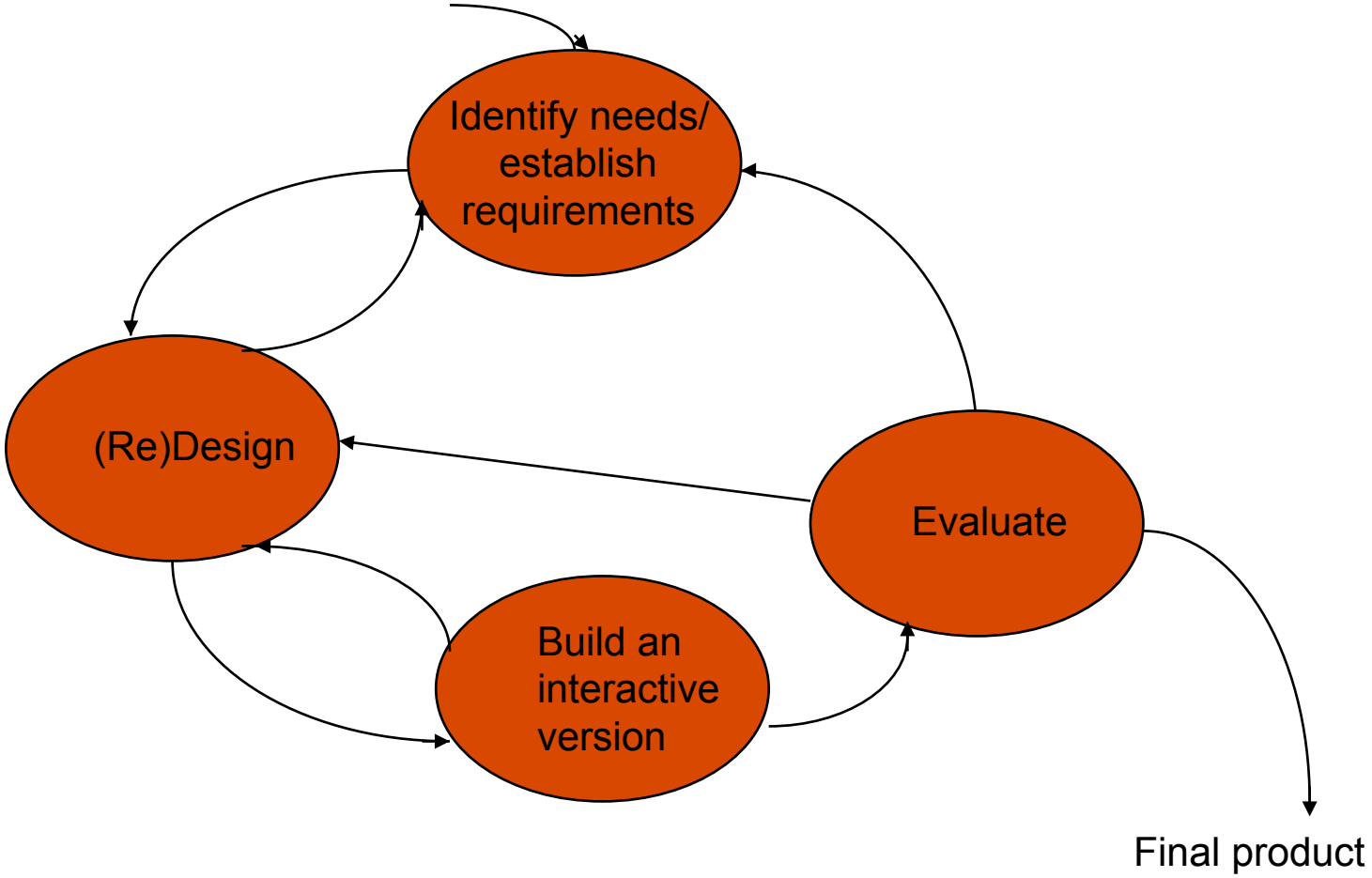
Where do alternatives come from?

- Humans stick to what they know works
- But considering alternatives is important to 'break out of the box'
- Designers are trained to consider alternatives, software people generally are not
- How do you generate alternatives?
  - —'Flair and creativity': research & synthesis
  - —Seek inspiration: look at similar products or look at very different products

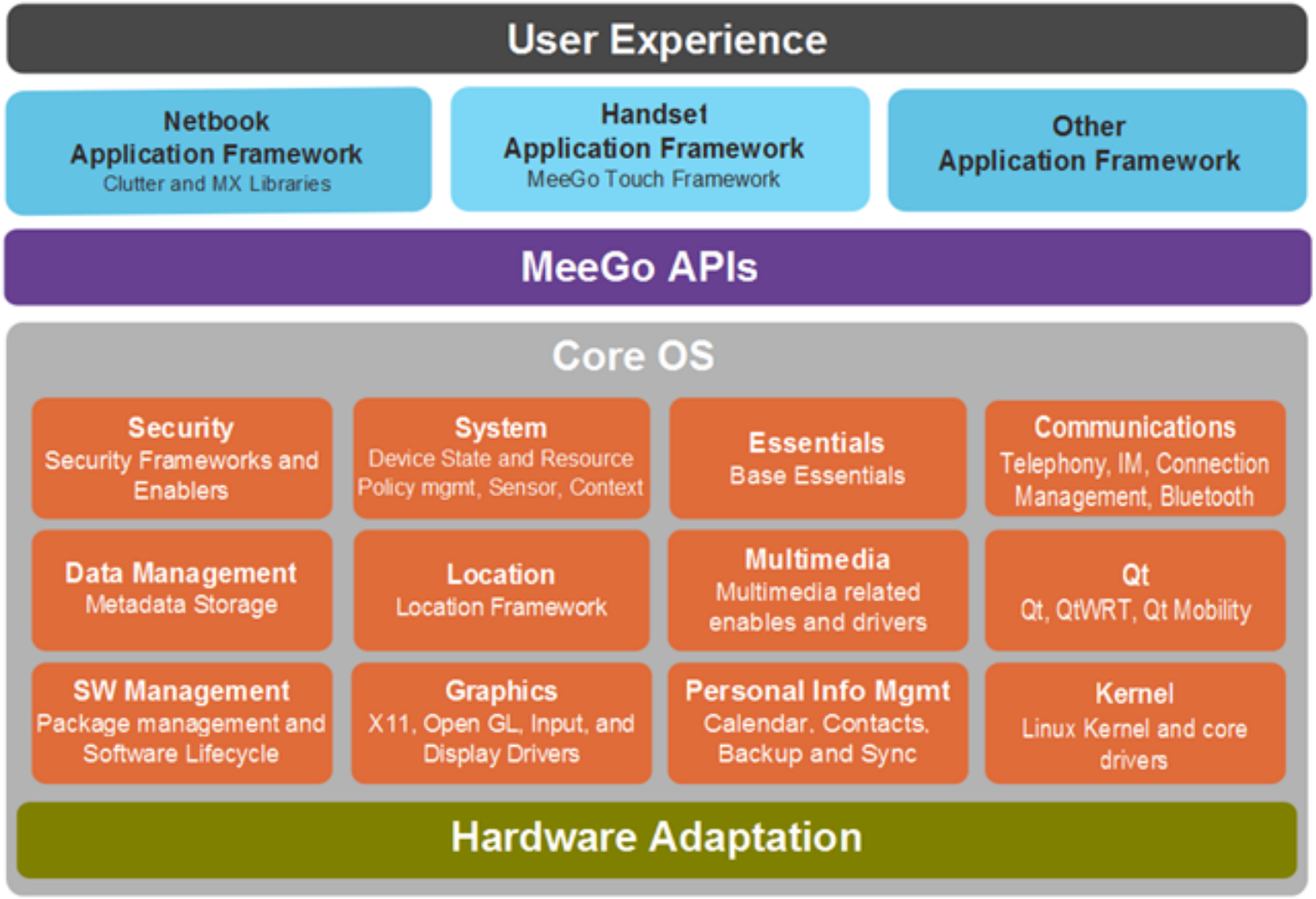
How do you choose among alternatives?

- Evaluation with users or with peers e.g. prototypes
- Technical feasibility: some not possible
- Quality thresholds: Usability goals lead to usability criteria (set early and checked regularly)
  - —safety: how safe?
  - —utility: which functions are superfluous?
  - —effectiveness: appropriate support? task coverage, information available
  - —efficiency: performance measurements

A simple HCI design model

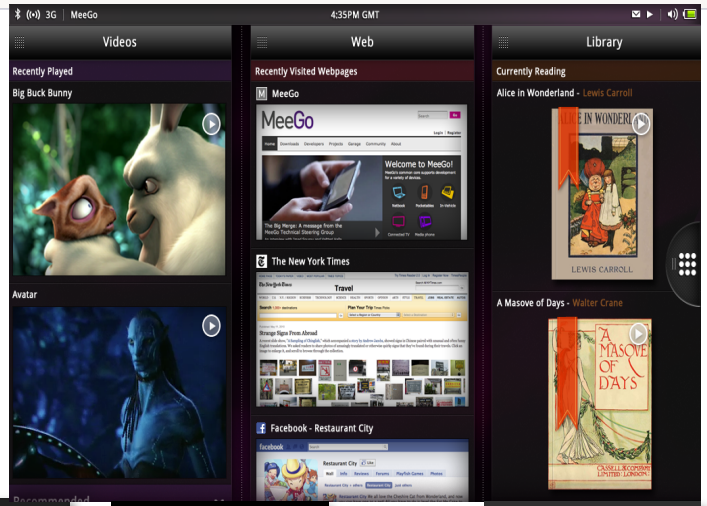


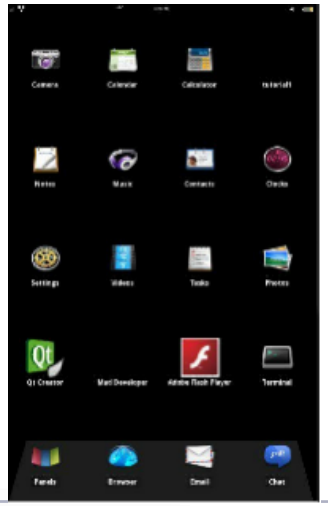
UX in MeeGo

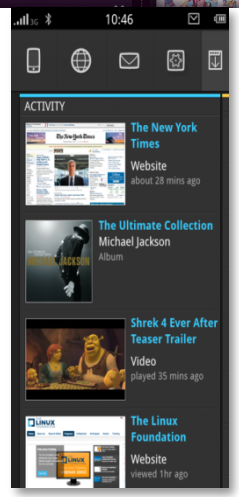

User Experience
Netbook
Application Framework
Clutter and MX Libraries
Handset
Application Framework
MeeGo Touch Framework
Other
Application Framework
MeeGo APIs
Core OS
Security
Security Frameworks and Enablers
System
Device State and Resource Policy mgmt, Sensor, Context
Essentials
Base Essentials
Communications
Telephony, IM, Connection Management, Bluetooth
Data Management
Metadata Storage
Location
Location Framework
Multimedia
Multimedia related enables and drivers
Qt
Qt, QtWRT, Qt Mobility
SW Management
Package management and Software Lifecycle
Graphics
X11, Open GL, Input, and Display Drivers
Personal Info Mgmt
Calendar, Contacts, Backup and Sync
Kernel
Linux Kernel and core drivers
Hardware Adaptation

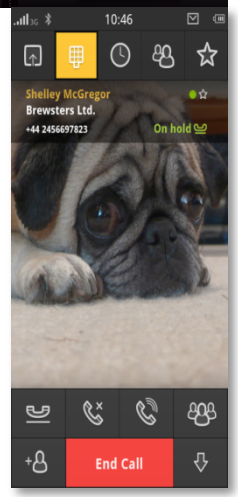
Differentiate-Able User Experiences

- Customizable Look and Feel
- Pre-integrated Apps and Services
- Full Internet
- Rich Media
- 3D Animation
- Application Stores



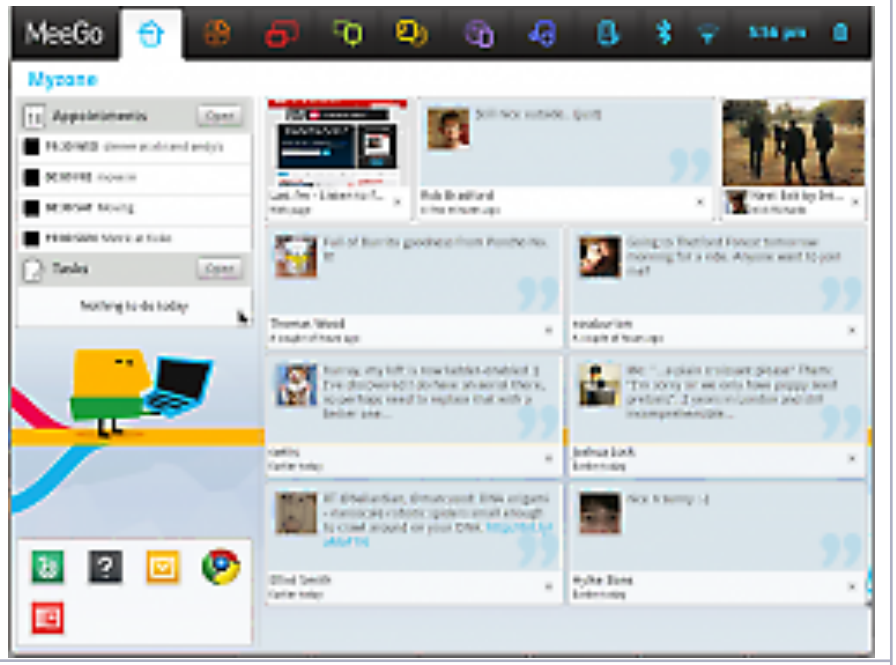






MeeGo for Netbooks

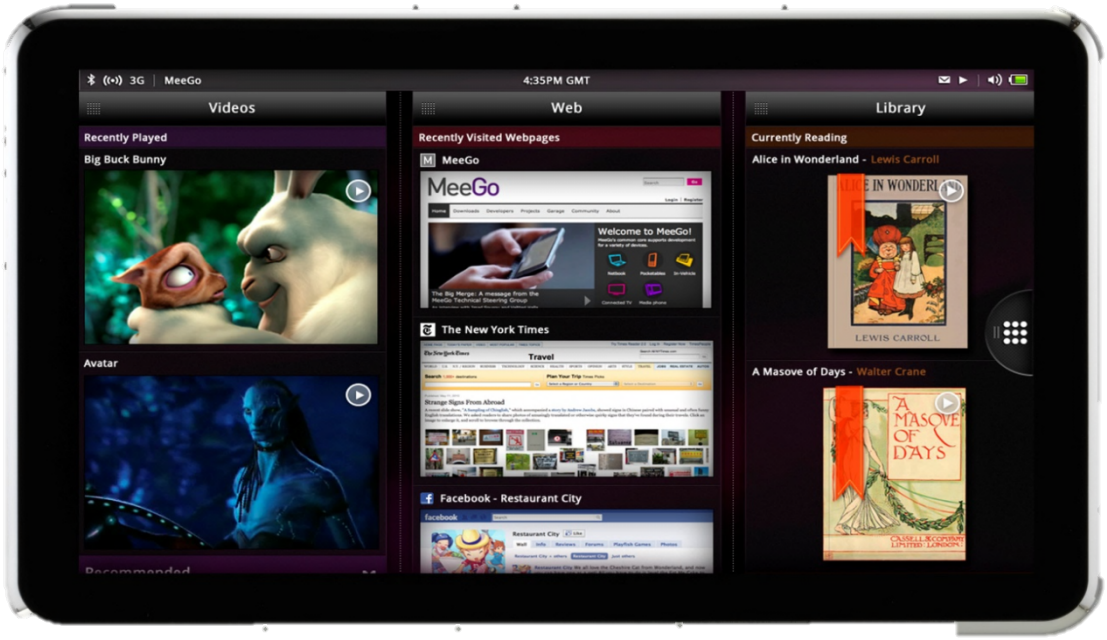
- Visually rich netbook user experience
- Touch support integrated on netbooks/nettops.
- Instant access to calendar, tasks, appointments, recently used files, and real-time social networking updates through the home screen.
- Aggregation of your social networking content.
- Power and performance optimization.
- multiple Languages support: Japanese, Korean, Chinese Simplified...



MeeGo for Handset

- The MeeGo based platform is designed to enable the application and services ecosystem for these mobile, rich internet and media-centric devices.

MeeGo for Tablet



engadget

01 June 2010

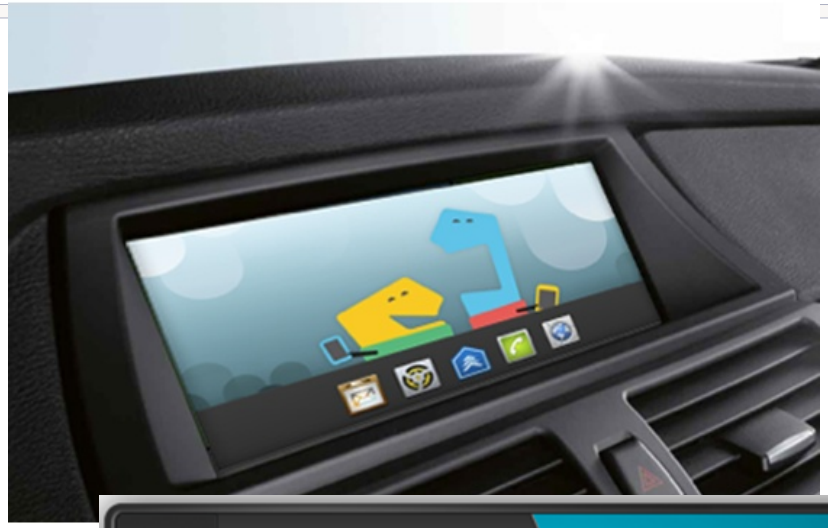
"…the most impressive thing may have just been MeeGo running on a 10-inch Moorestown Quanta Redvale tablet….To say we're impressed with the "pre-alpha" version of the software is a huge understatement."

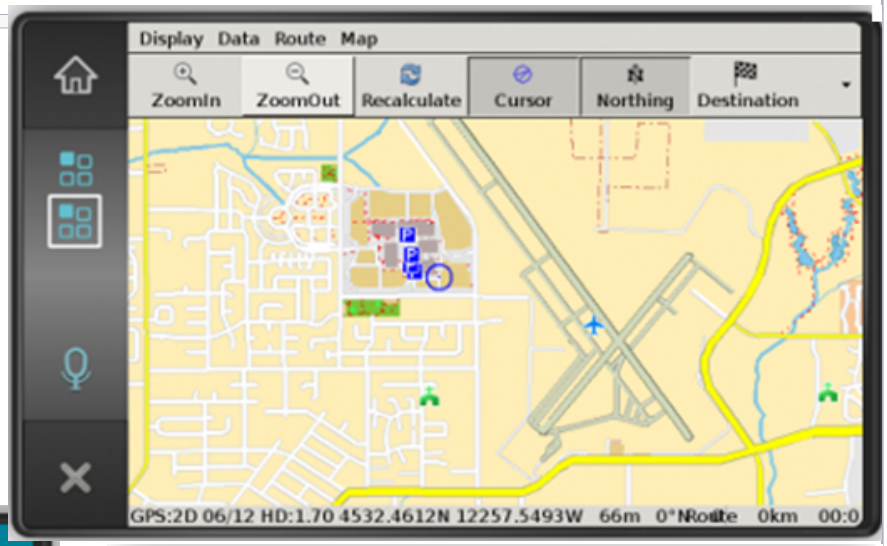
liliputing

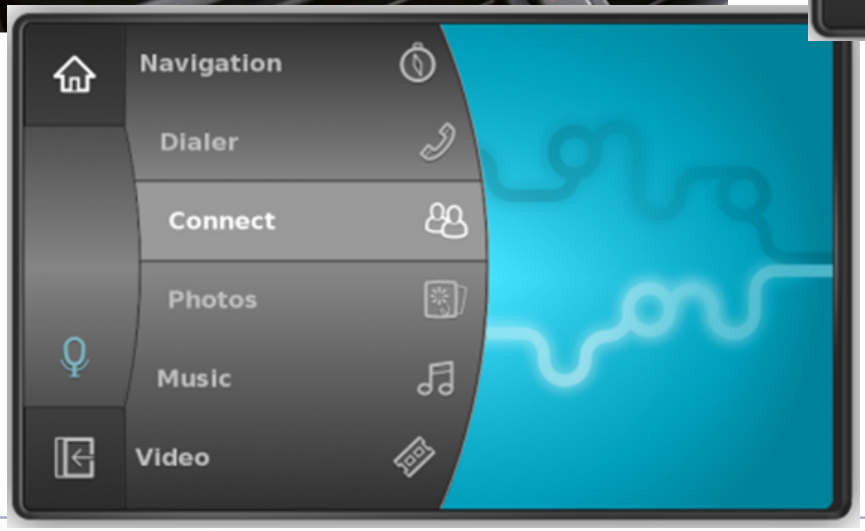
01 June 2010

"I have to say, I'm pretty excited about the prospect of tablets running MeeGo. ...the user interface really looks like it was *designed* to be touched. It has nice big icons and every app launches in full screen the way it would on a smartphone."

MeeGo IVI



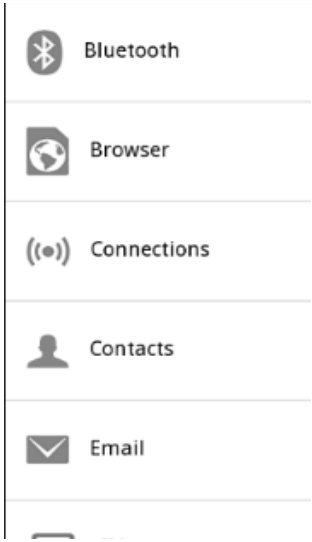

Display Data Route Map
ZoomIn
ZoomOut
Recalculate
Cursor
Northing
Destination
GPS:2D 06/12 HD:1.70 4532.4612N 12257.5493W 66m 0°N
0km 00:0

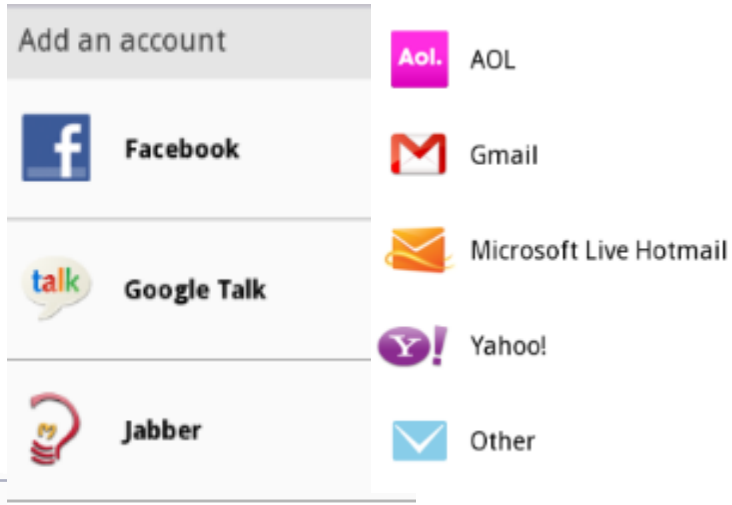

Navigation
Dialer
Connect
Photos
Music
Video


611
!! Unknown Caller
!! Hold
!! Mute
!! Spea...
!! Merg...
!! End Call

MeeGo UX Design Principles

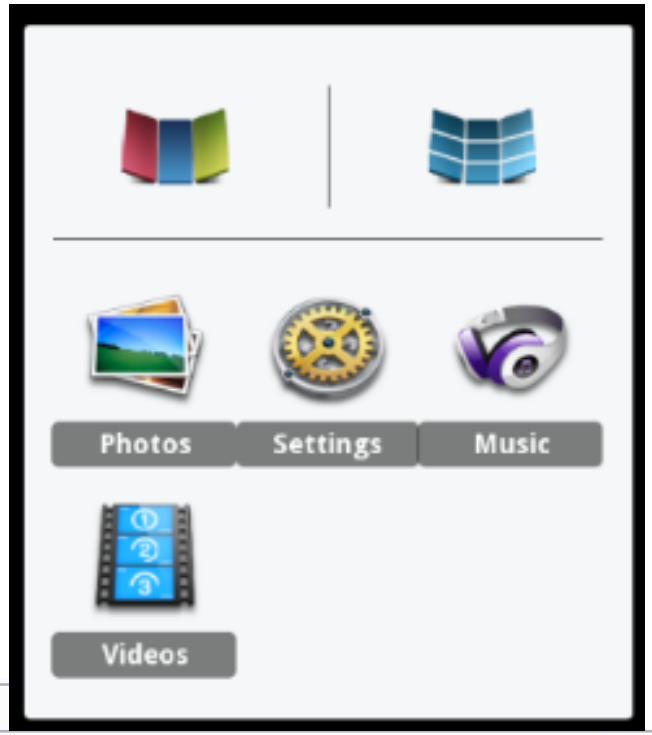
- Connected, Vibrant and Alive
  - Connect to the web all the time
  - Feel alive with relevant and optimized activity
  - Consider power consumption
  - Make the device a true life hub

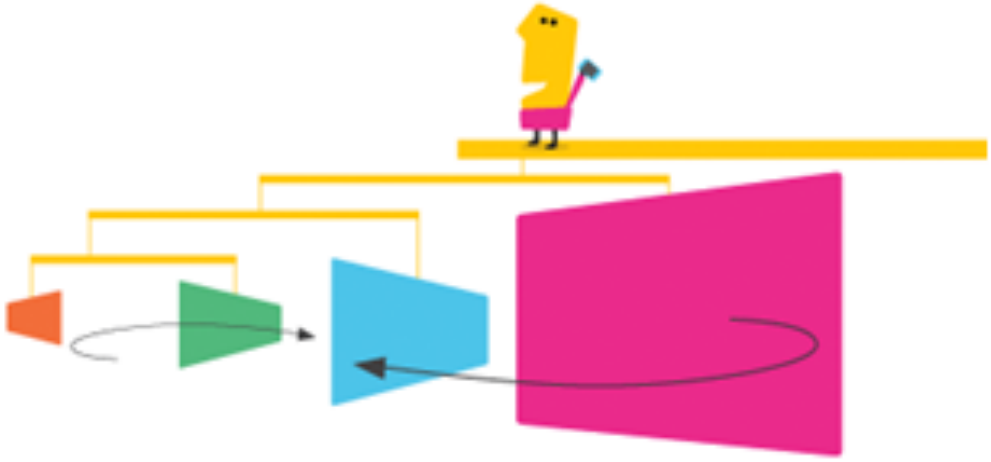




MeeGo UX Design Principles

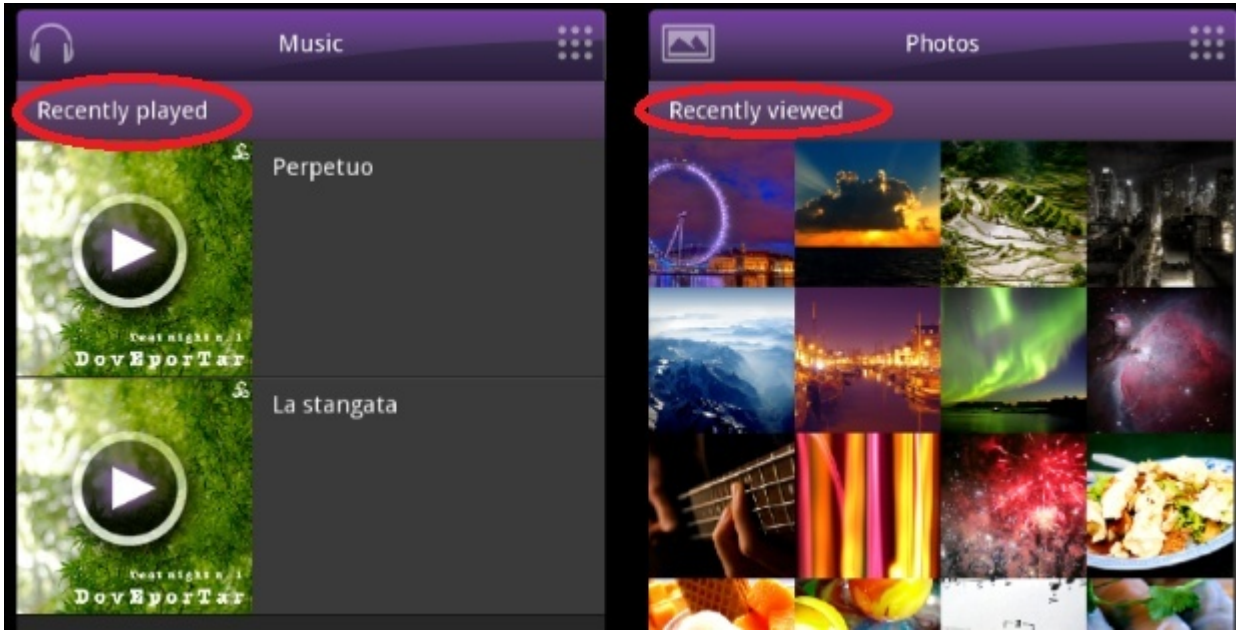
- Task Switching & Multi-tasking
  - Allows people to quickly move around running applications





MeeGo UX Design Principles

- Adaptive & Intelligent
  - MeeGo is intelligent, and adaptive
  - Learn from the users habits



MeeGo UX Design Principles

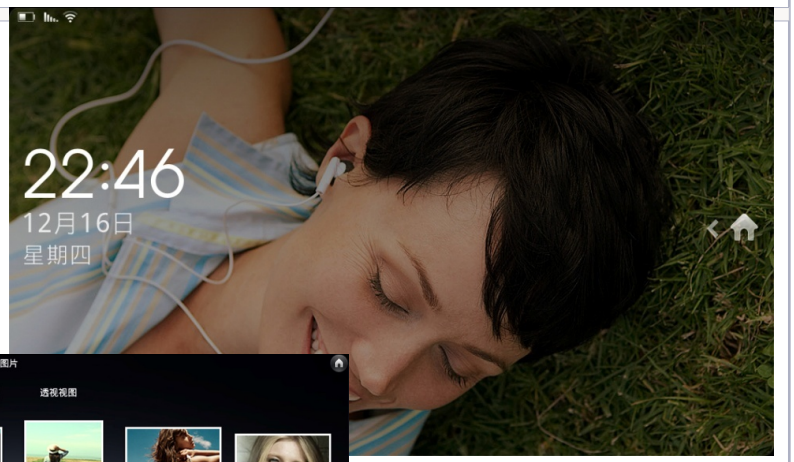
- Responsive
  - Give immediate feedback to the user's actions
  - Visual, audio and haptic feedback must feel that they are in perfect synch.

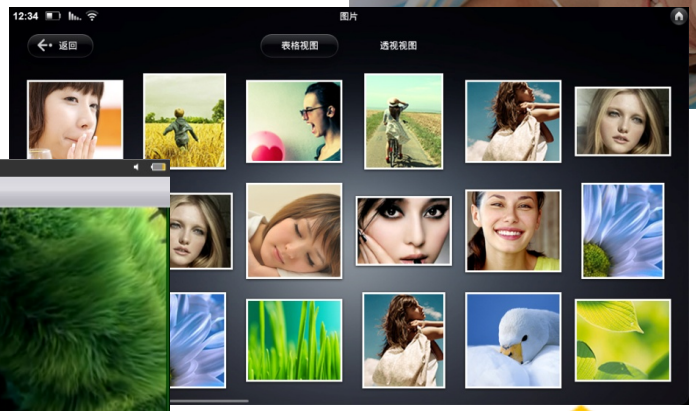
MeeGo UX Design Principles

- Getting the basics right
  - Make every day life things a joy and never be a distraction or overly elaborate

MeeGo UX Design Principles

- Simply beautiful
  - Be beautiful and simple of visual, aural and haptic feedback
  - Be consistent of visual and aural effects





MeeGo UX Design Principles

- Plug-ins & Framework support
  - Provide frameworks which improve the user experience, ease and speed up the development

What is customization

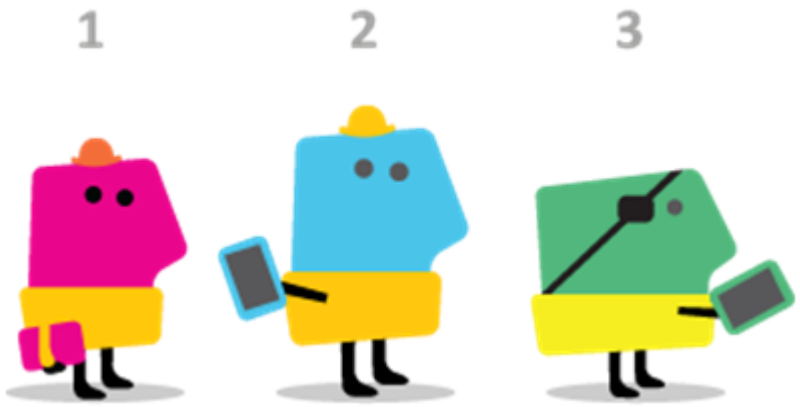
- MeeGo UX customization is modifying device by setting the value of a parameter, exchanging files, or modifying a setting within a given predefined range
  - Customization relates to offered services, available applications, widgets, and plug-ins, as well as device usage and visual appearance.
  - Customization does not require significant programming effort on the part of the handset provider.
  - End user personalization is not considered customization.

What is customization

- Customization options related to
  - Visual appearance
  - Experience of using the device
  - Motion graphics
  - Sounds
  - Incorporating your services and plug-ins

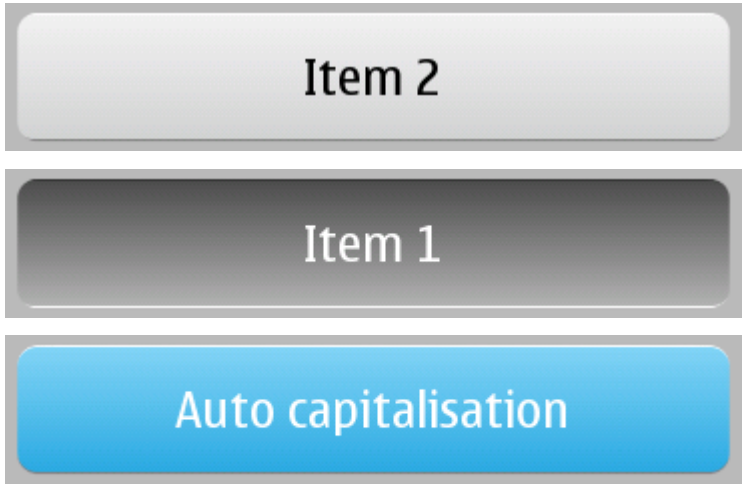
What is customization

- Customization can be divided into three levels:
  - Customizing the Visual Appearance
  - Customizing the Experience
  - Customizing Plug-ins and Applications



Customizing the Visual Appearance

- Colors
  - The component graphics are SVG vector images which can be changed with Inkscape or Illustrator CS4
  - The text colors have a global logic, which is changed from the constants.ini text file



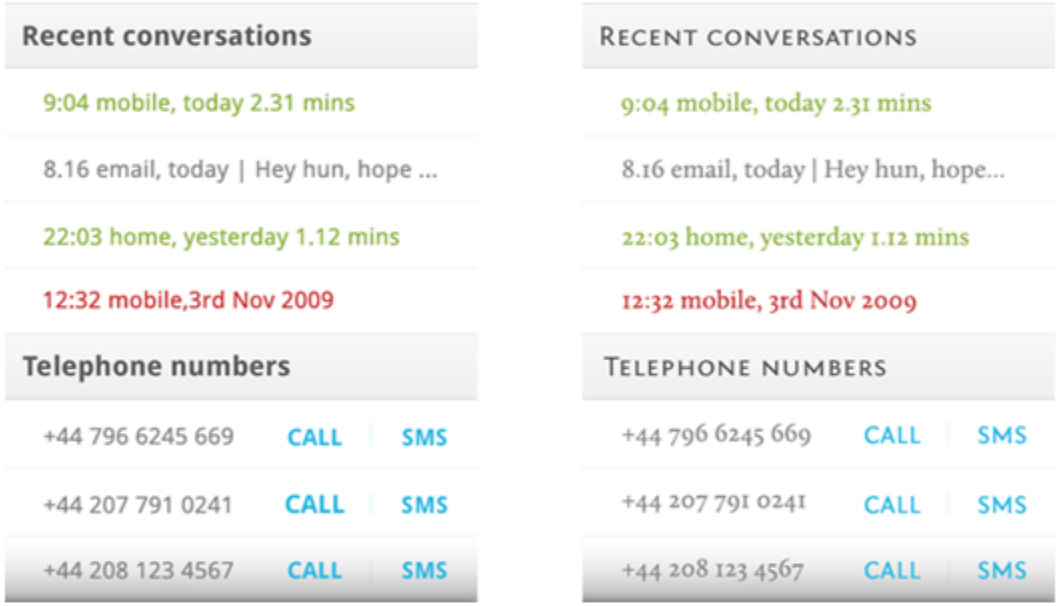
Normal State

Pressed State

Selected State

Customizing the Visual Appearance

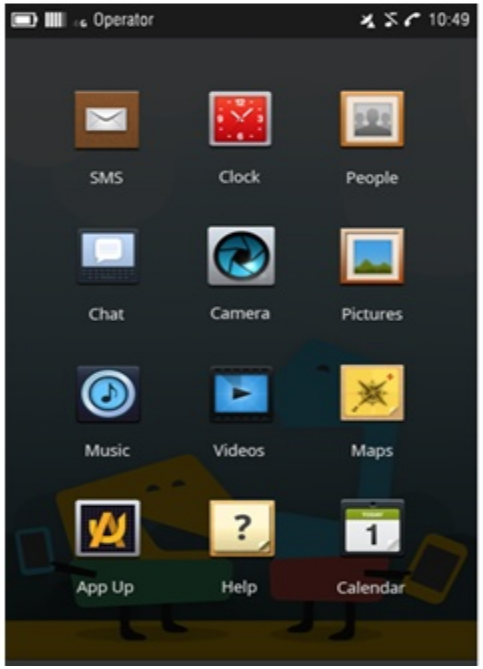
- Fonts
  - The global font families and their sizes are customizable
  - The font name and size can be changed in the constants.ini document

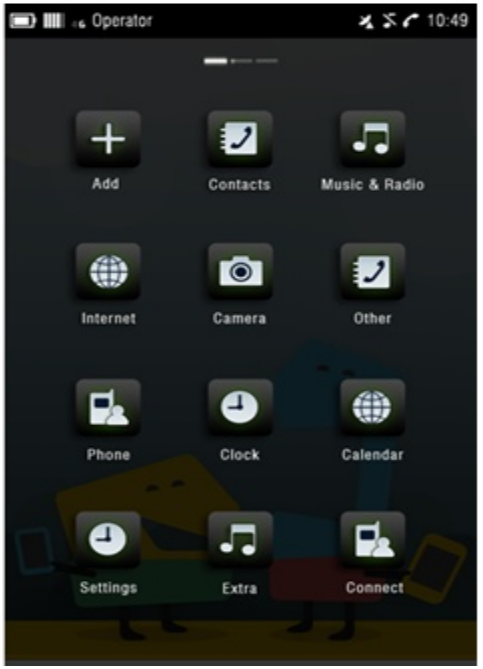


example of change font

Customizing the Visual Appearance

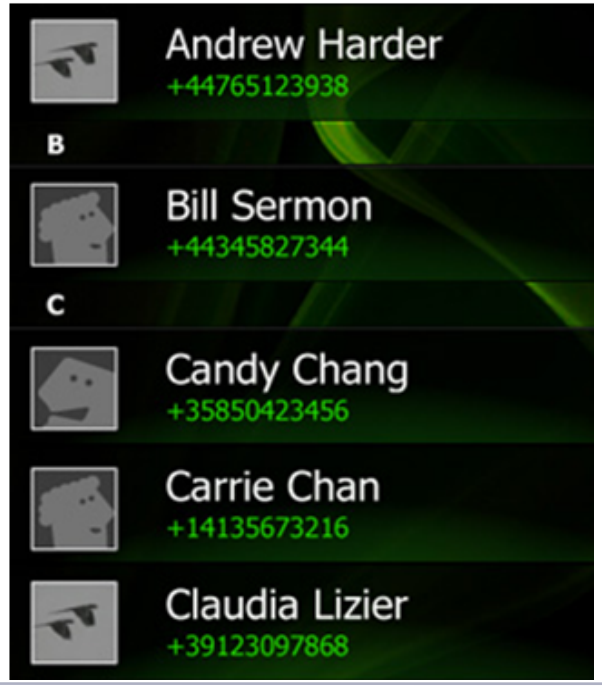
- Icons
  - Any icons on the device can be customized, either framework or application icons. Only the changed icons are affected.
  - The icons are SVG vector files stored in the Icons folder





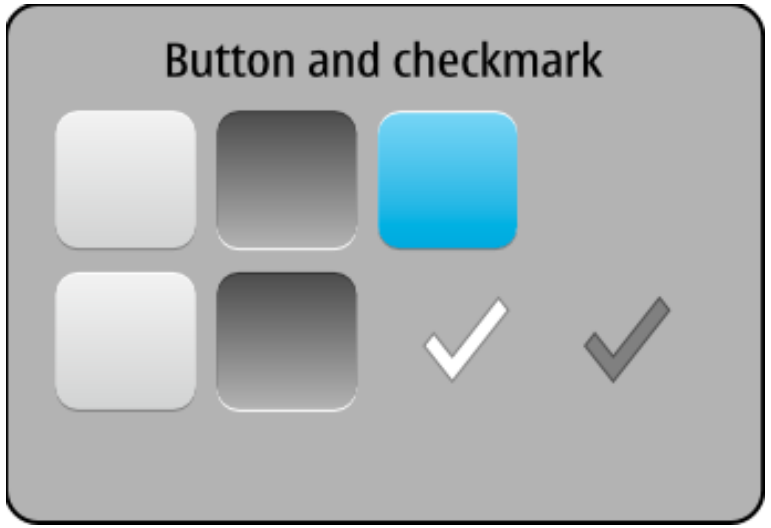
Customizing the Visual Appearance

- Application background
  - The application background is in the Images folder which is called duiapplicationpage-background.png



Customizing the Visual Appearance

- Selected state gradient
  - The selected state color is visible in many places on the interface, in buttons, switchers, lists, notifications, and other interactive elements.
  - Modify colored gradient for svg graphic file to customize selected state gradient



Customizing the Visual Appearance

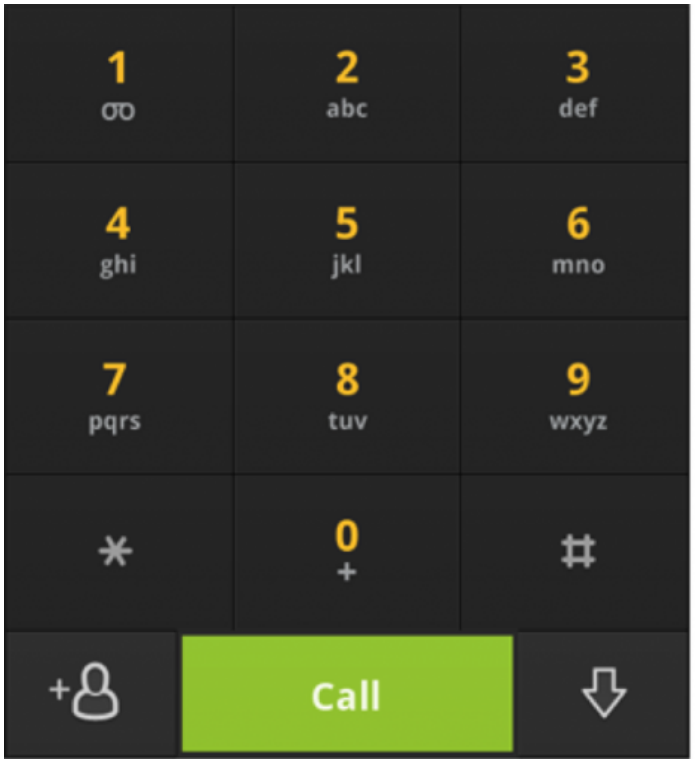
- Gradients of all button states
  - Customizing gradients is a safe way to change the look and feel of the buttons and other elements

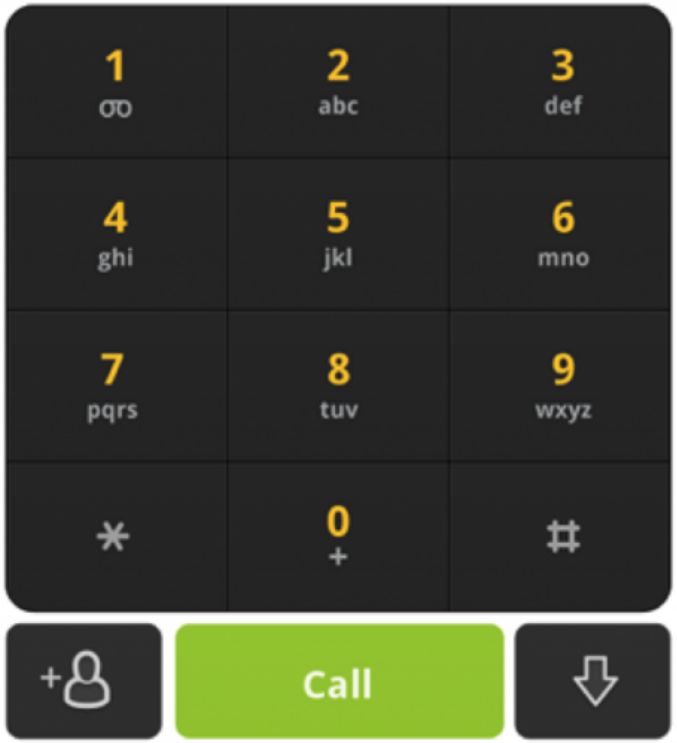
Customizing the Visual Appearance

- Transparency
  - MeeGo offers the possibility to use transparency in graphics.
  - Transparency needs to be carefully tested in the real device environment, as the final effect cannot always be estimated beforehand in the graphics tool

Customizing the Visual Appearance

- Modifying component styles by CSS files



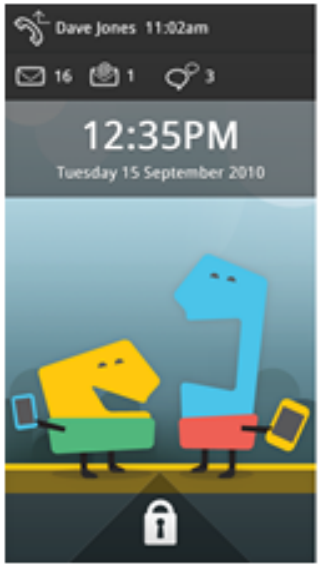


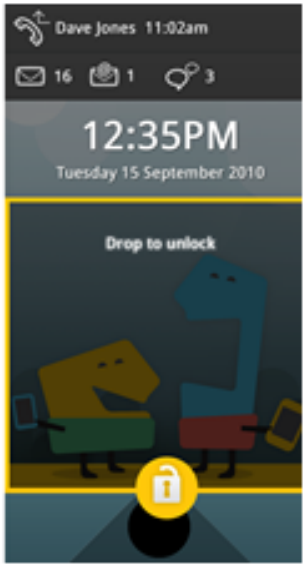
Customizing the Experience

- Use flow
  - Customizing the use flows has a powerful impact on the device user experience
  - Use flows include mainly
    - Lock screen and unlock sequence
    - Home screen
    - Launcher
    - Quick launch bar
    - Navigation bar

Customizing the Experience

- Use flow - Lock screen and unlock sequence
  - The lock screen offers two alternatives:
    - Date/time at the top and unlock mechanism at the bottom
    - Date/time and unlock mechanism at the top





The position of the lock; The lock screen button and drop area;
Update the colors, lock-button style, and transparency

Customizing the Experience

- Use flow - Home screen
  - The home screen can have two different default layouts, a carousel, or a grid
  - The carousel view displays the running applications in a carousel-like format
  - In grid view the running applications are displayed in a grid

carousel view **grid view**

Customizing the Experience

The size of the cards, how much they overlap, how much the cards zoom, and even transition speed and rotation can be modified.

The amount of rows and columns can be changed, as well as the spacing between the cards. The page indicator is the same as in the carousel view

grid view

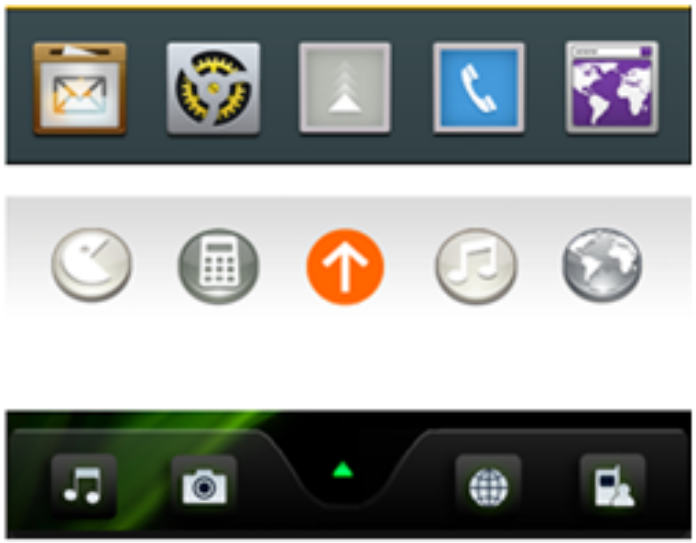
Customizing the Experience

- Use flow - Launcher
  - The launcher features all the applications available on the device
  - Customization of the theme and icon graphics; The size and spacing of the icons; The icons press and release feedback graphics, sound, and haptics; the text size, color, and position of launcher buttons

different launcher customizations

Customizing the Experience

- Use flow - Quick launch bar
  - The quick launch bar is situated on the bottom of the launcher, switcher, and other home views
  - Customization of graphic background and its size; The margins between icons, icon sizes, text color, and feedbacks



Customizing the Experience

- Use flow - Navigation bar
  - The navigation bar is shown on top of each application
  - By customizing it, you can easily change the look and feel of applications

Customizing the Experience

- Motion graphics
  - Motion graphics refers to the transitions shown when interacting with the device
  - keep the motion graphics for similar events consistent

Customizing the Experience

- Sounds
  - Sounds are the strongest non-visual differentiators of the device.
  - The table below lists the most relevant sounds to customize
    - Press
    - Release
    - Cancel
    - Ringtone
    - SMS
    - Email
    - IM

Customizing the Experience

- Haptics
  - Haptics can be used to provide the end user with the information that they have performed an action
  - Provide a file describing the haptic and change the corresponding parameter in the related CSS file to customize haptics
  - The table below lists the most relevant sounds to customize
    - Press
    - Release
    - Cancel

Customizing Plug-ins and Applications

- Plug-ins
  - Customization options are the usage of the plug-ins as available
  - provide the binaries for the new plug-ins as part of the configuration package

Customizing Plug-ins and Applications

- Application
  - Application customization can be done depends on the default set of applications available
  - The placement of application icons in the Launcher can be customized
  - To customize an application, modify the corresponding parameters in the appropriate configuration package

MeeGo Graphics

- EGL
- OpenGL ES
- QtOpenGL

EGL

- The Khronos Native Platform Graphics Interface (EGL) is an open standard developed by the Khronos Group.
  - An interface between the platform window system and rendering APIs such as OpenGL and OpenVG
  - Makes it possible to use several rendering APIs together
  - Can be used for accelerated mixed-mode 2D and 3D rendering

- Following is an example showing EGL display

EGL API

- eglGetDisplay
  - returns a EGLDisplay data type.

```
1  EGLDisplay display;
2
3  display = eglGetDisplay(EGL_DEFAULT_DISPLAY);
4
5  if (eglInitialize(display, NULL, NULL))
6  {
7      // Proceed with your code.
8  }
```

  - A constant EGL_DEFAULT_DISPLAY is always implemented by the vendors to return their default display.

EGL Context

EGL CONTEXT

EGLContext eglCreateContext(EGLDisplay display, EGLConfig config, EGLContext shareContext, const EGLint* attribList)

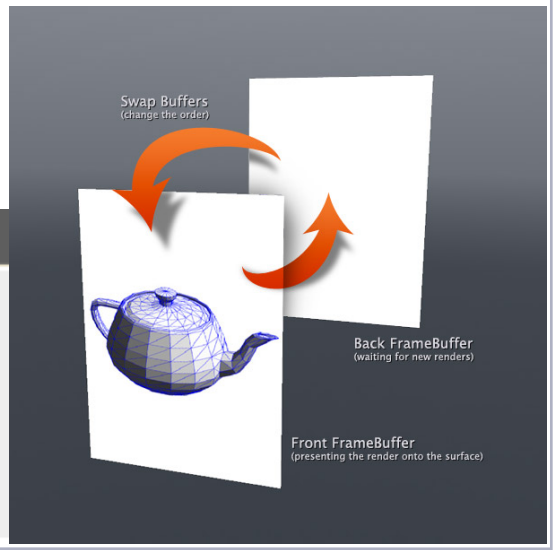
- **display**: The previously get display.
- **config**: The previously set configuration.
- **shareContext**: Usually is EGL_NO_CONTEXT to share no context.
- **attribList**: To OpenGL ES, this parameter determines which version of OpenGL ES will be used. Value 1 represent a context of OpenGL ES 1.x and a value of 2 represent a context of OpenGL ES 2.x.

EGLBoolean eglmakeCurrent(EGLDisplay display, EGLSurface draw, EGLSurface read, EGLContext context)

- **display**: The display obtained previously.
- **draw**: The surface obtained previously.
- **read**: The same value as draw.
- **context**: The context to be the current.

Render with EGL context

- Once the EGL knows about the windowing system and has a context to know about OpenGL application, we can make our render's output be presented onto the desired surface.
- While EGL present one frame buffer to your desired surface, the other one stay on back waiting for a new render output.



SWAP THE EGL'S BUFFERS

EGLBoolean eglSwapBuffers(EGLDisplay display, EGLSurface surface)

- **display**: The display obtained previously.
- **surface**: The surface obtained previously.

OpenGL ES

- OpenGL ES is a cross-platform API for 2D and 3D graphics on embedded systems.
  - Defines relative to the OpenGL specification
  - Emphasizes hardware acceleration of the API
  - Features improved performance, image quality and functionality
  - Reduces memory bandwidth usage in order to save power

QtOpenGL

- OpenGL only deals with 3D rendering and provides little or no support for GUI programming issues.
- The user interface for an OpenGL application must be created with another toolkit, such as Motif on the X platform, MFC under Windows, or Qt on both platforms.
- The Qt OpenGL module makes it easy to use OpenGL in Qt applications.
  - It provides an OpenGL widget class that can be used just like any other Qt widget.

Examples

- Hellogl
- 2dpainting
- Overpainting