# openmoko

NEOLOGICS, FEBRUARY 2007

# The Current Phone Problem.

| | PC | Mobile Phone |
|---|---|---|
| Service | Google, Yahoo, AOL, Windows Live, YouTube | |
| Apps | Web browser. All kinds of vertical niche applications. | No *Open* Solution |
| GUI | Common “desktop” paradigm | |
| Input | Generic: usually keyboard, mouse, and monitor | Specialized: keypad, buttons, and inconsistent (and often limited) screen space |
| HW | x86 (Intel, AMD, VIA) | Lots of different platforms |

# But if we view this merely as an engineering problem to be solved...

- Then we WILL create a mobile phone that mimics a PC. We can do better than a 1960s vision.
- We would solve the problem, but we will fail to create new forms of computing.
- That's winning a battle, but losing the war.
- So how do we create a new form of computing?
- Who is in charge?  Or better yet, WHAT is in charge?

# How to be God.

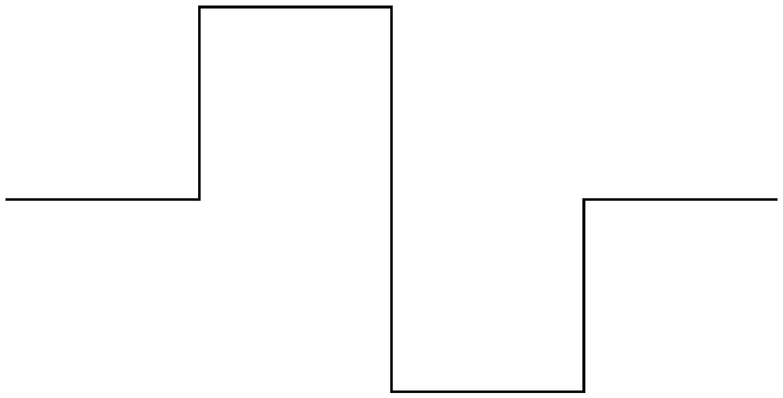
- The KEY to making complex, neoforms appear from simple systems is:
  - Access to the Building Blocks. The Amino acids of the systems. The Atoms of Molecules.
  - Freedom to WRITE new rules of combination.
- Let’s take an example of starting small...

Big fleas have little fleas
on their backs to bite them,
and little fleas have lesser fleas,
and so ad infinitum.

What happens when you start small and iterate?

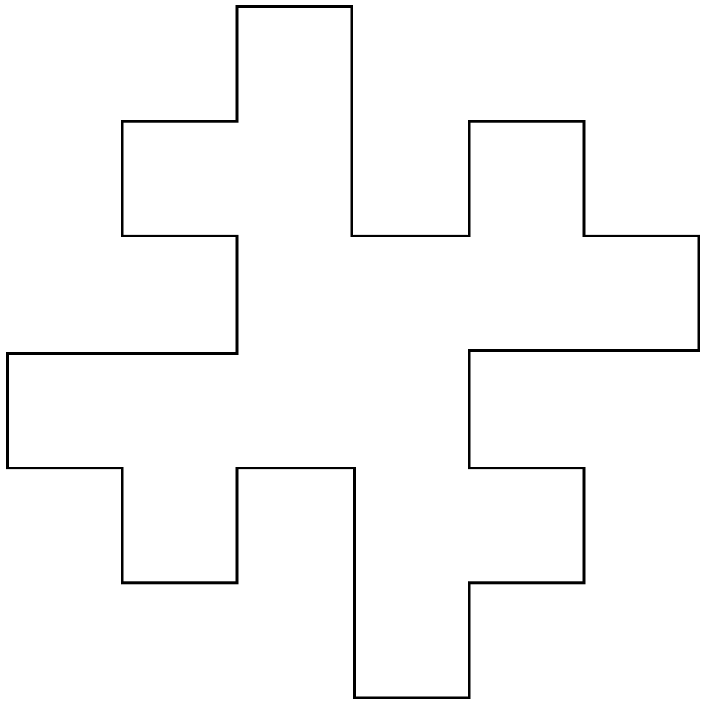
# It all starts small...

- Sometimes it’s the smallest, most simple concepts that work best.
- So take something small and begin...



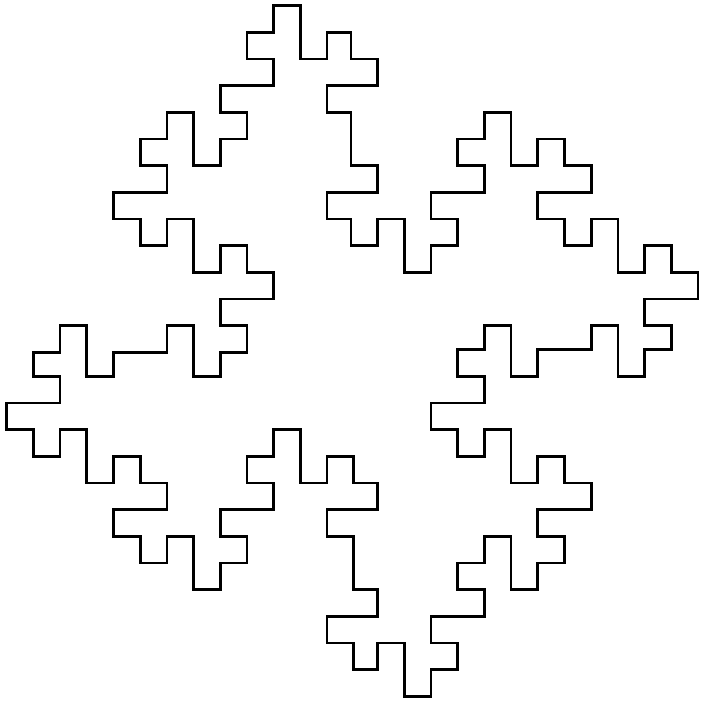
# Then iterate.

- The first iteration interpreted graphically looks like this:



# And do it again.

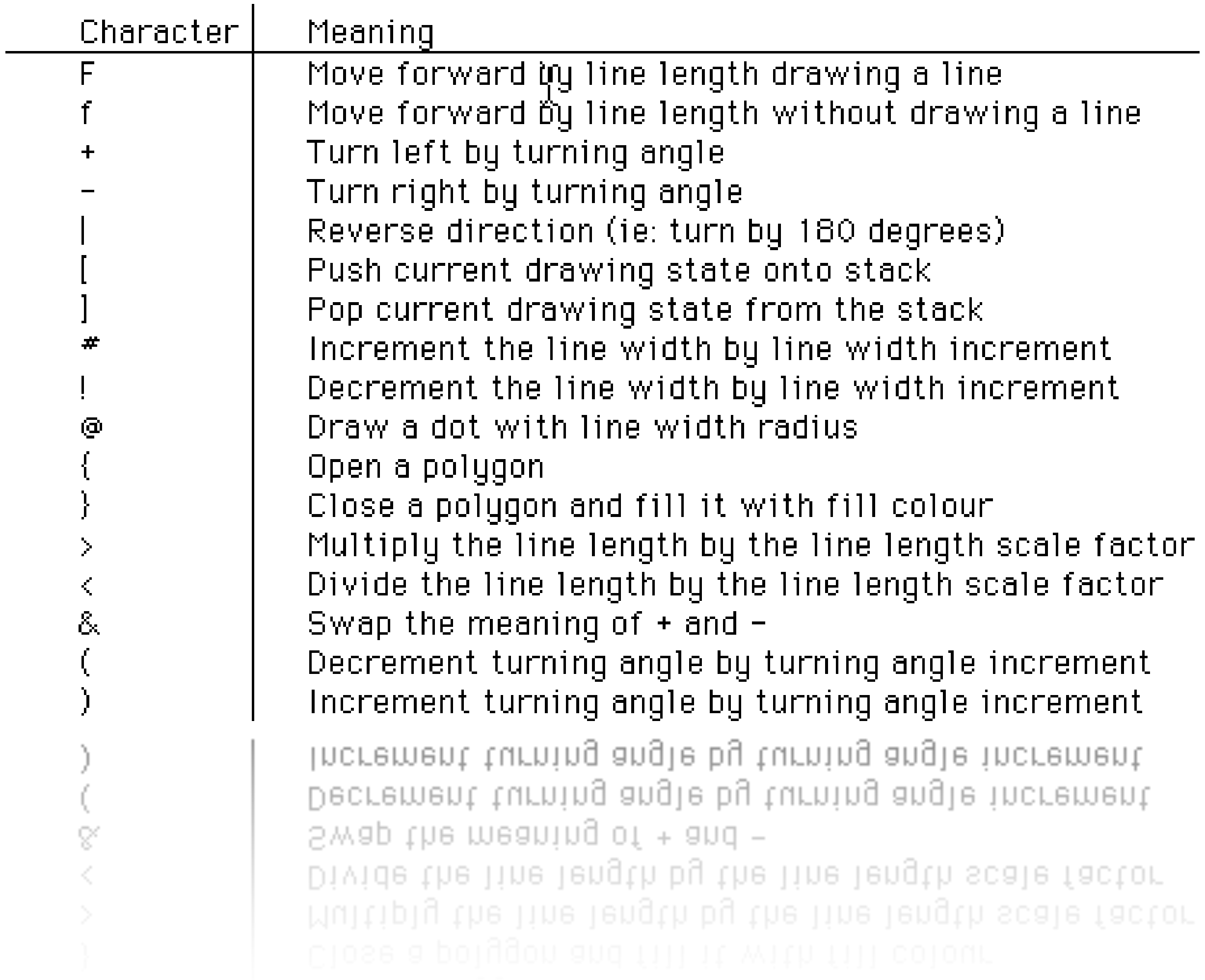
- The next iteration interpreted graphically might look something like this:



# Ok. Now what?

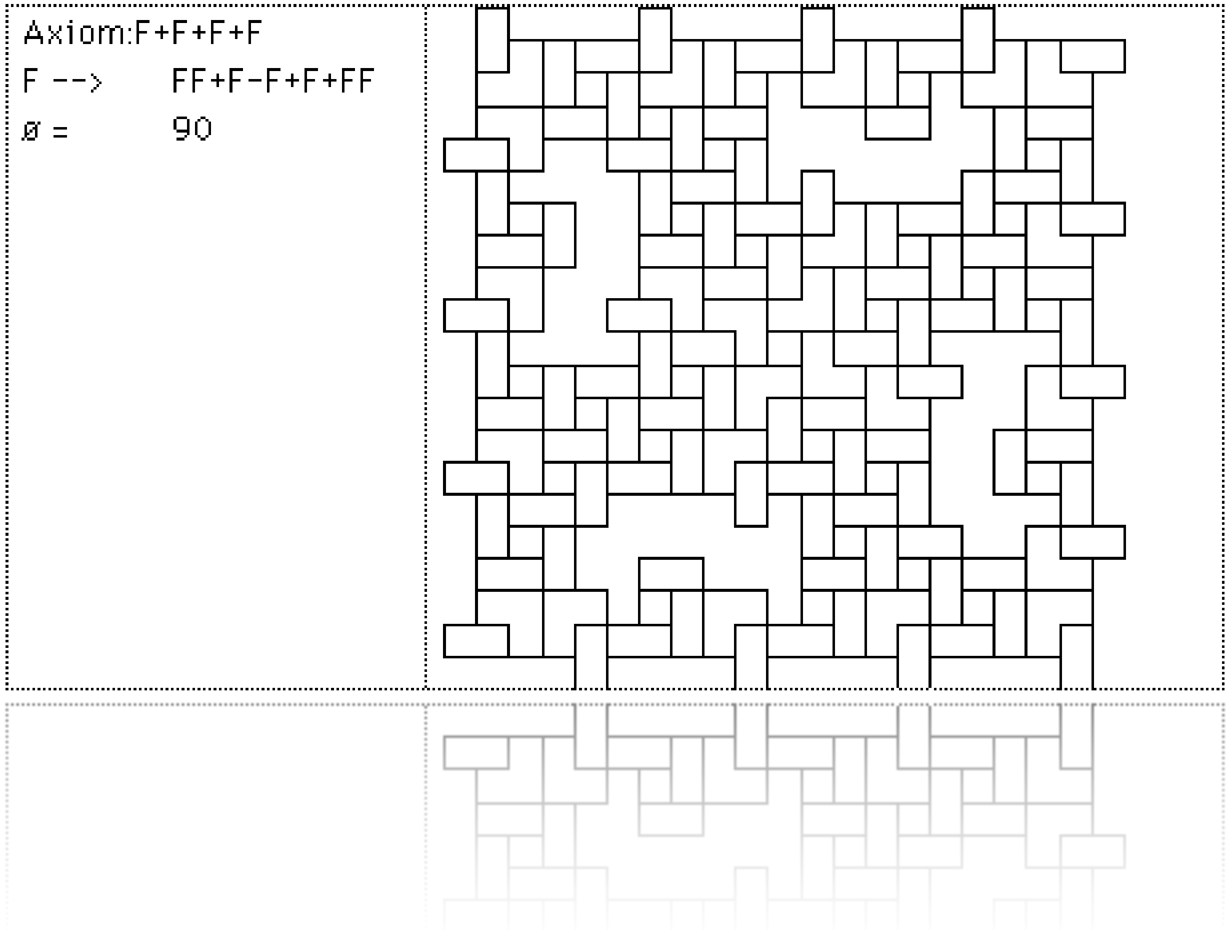
We have some basic building blocks in place...

# Let’s Define Some Rules...



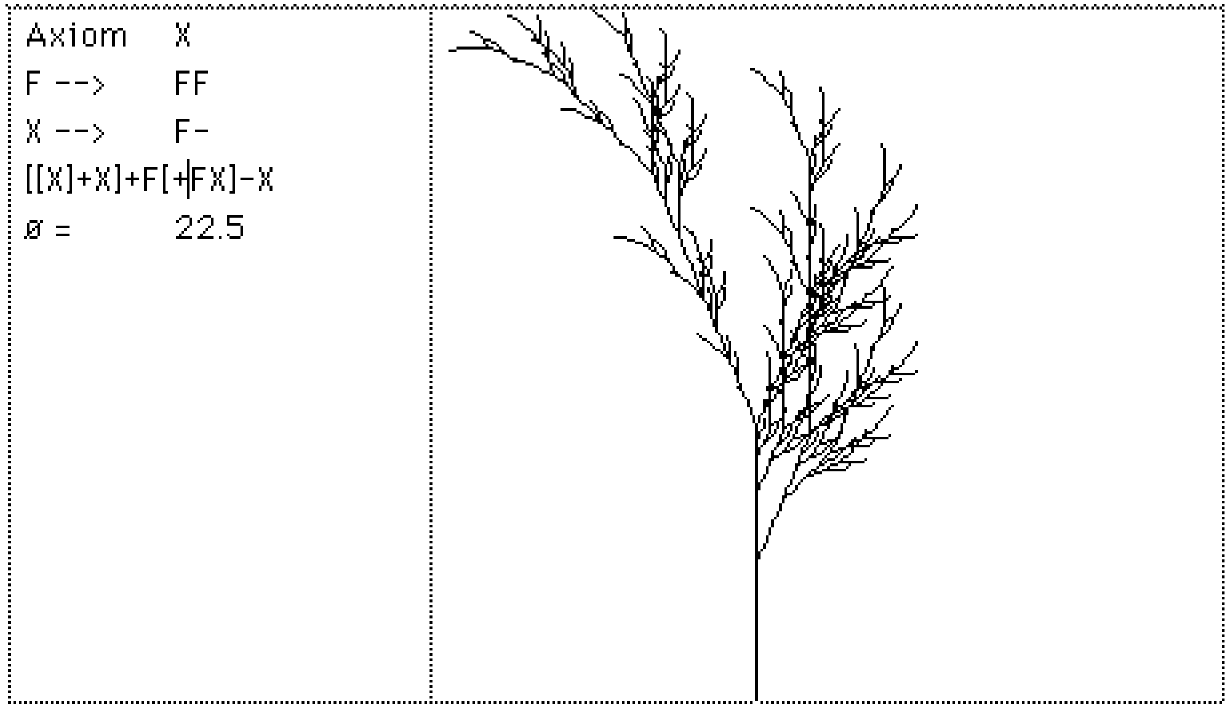
| Character | Meaning |
|---|---|
| F | Move forward by line length drawing a line |
| f | Move forward by line length without drawing a line |
| + | Turn left by turning angle |
| - | Turn right by turning angle |
| \| | Reverse direction (ie: turn by 180 degrees) |
| [ | Push current drawing state onto stack |
| ] | Pop current drawing state from the stack |
| # | Increment the line width by line width increment |
| ! | Decrement the line width by line width increment |
| @ | Draw a dot with line width radius |
| { | Open a polygon |
| } | Close a polygon and fill it with fill colour |
| > | Multiply the line length by the line length scale factor |
| < | Divide the line length by the line length scale factor |
| & | Swap the meaning of + and - |
| ( | Decrement turning angle by turning angle increment |
| ) | Increment turning angle by turning angle increment |

# And You Can Do This:



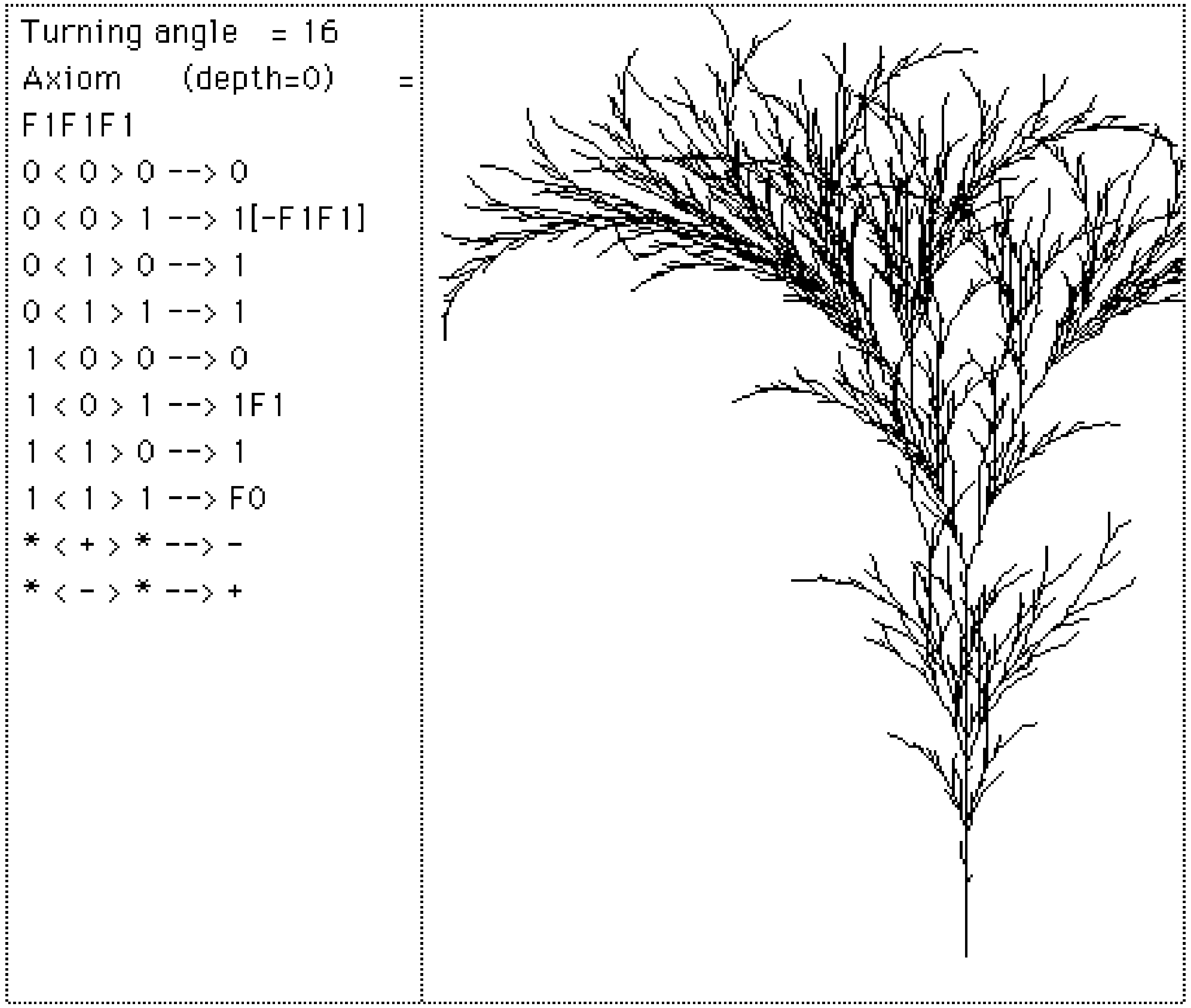
```
Axiom:F+F+F+F
F -->    FF+F-F+F+FF
ø =      90
```

# Even this!



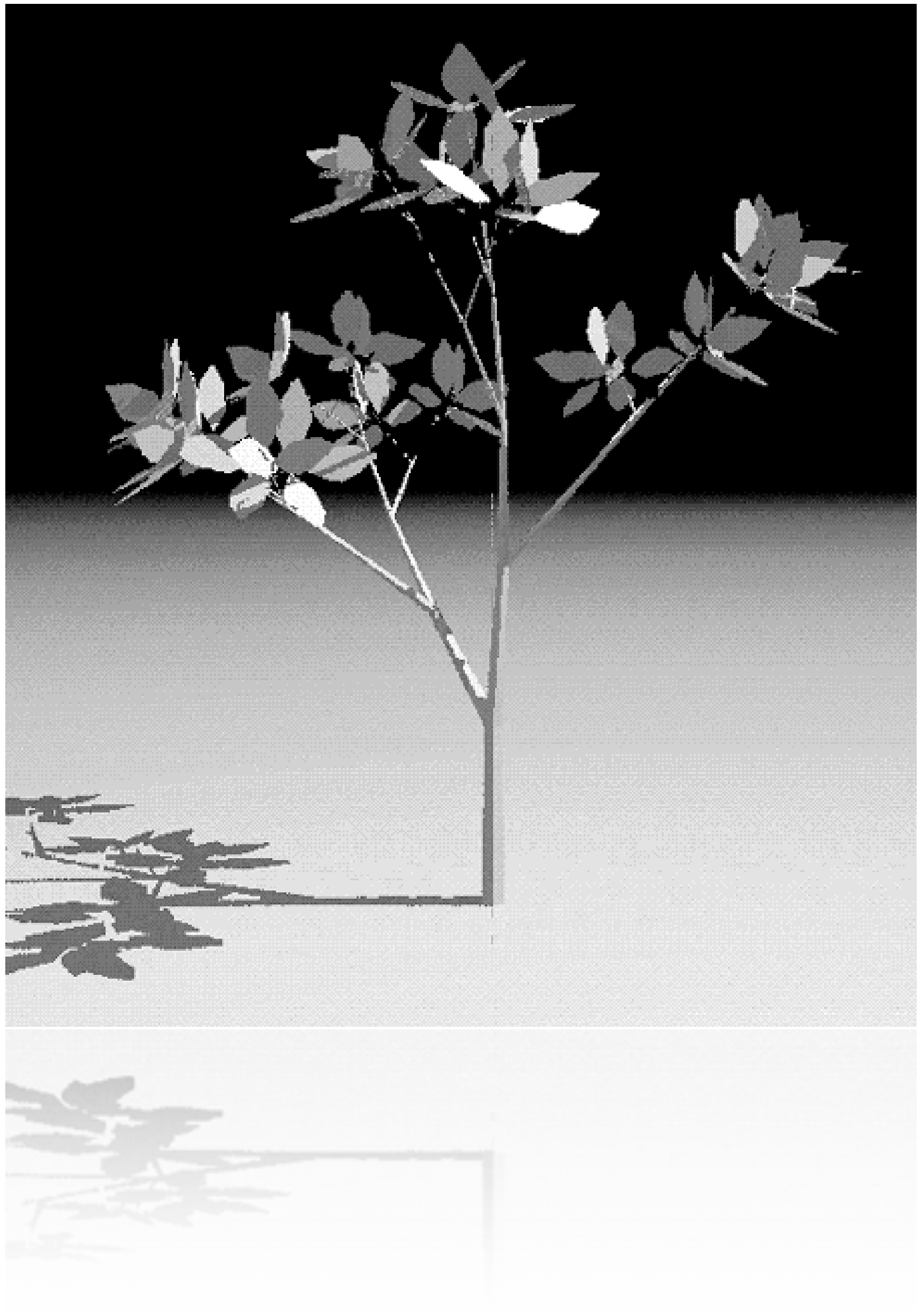
Axiom X
F --> FF
X --> F-
[[X]+X]+F[+FX]-X
ø = 22.5

# Does This Look Familiar?



```
Turning angle   = 16
Axiom     (depth=0)     =
F1F1F1
0 < 0 > 0 --> 0
0 < 0 > 1 --> 1[-F1F1]
0 < 1 > 0 --> 1
0 < 1 > 1 --> 1
1 < 0 > 0 --> 0
1 < 0 > 1 --> 1F1
1 < 1 > 0 --> 1
1 < 1 > 1 --> F0
* < + > * --> -
* < - > * --> +
```

# Now We’re Done.



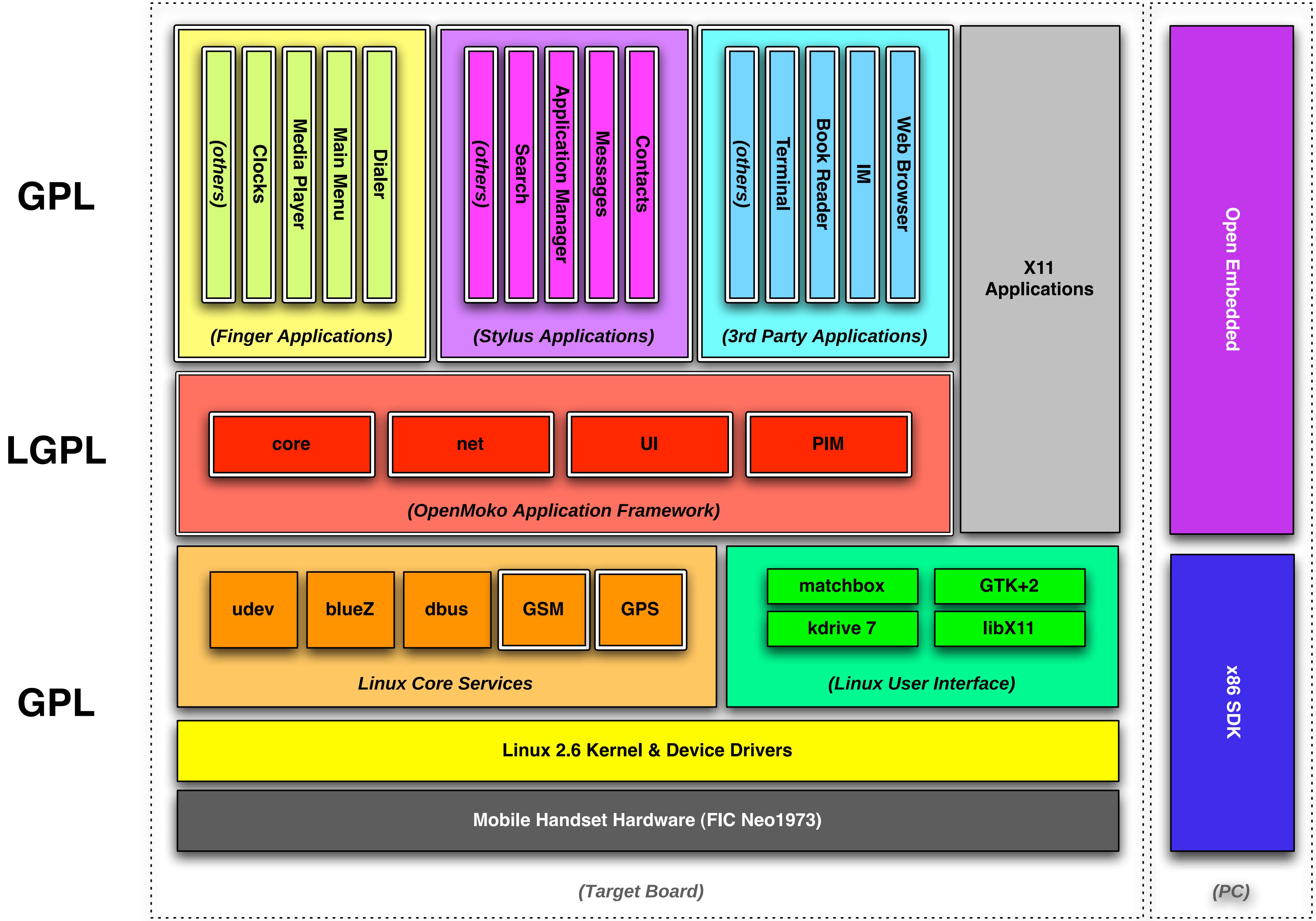
**Closed systems lead to controlled predictable evolution. NOT new species. Not Neos. Not punctuated equilibriums. Neos initially look like genetic errors. Mistakes. But they survive and outperform.**

**If we just recreate the PC on the Phone, it will be just another Flea. We need new species...**

# Standing on each other’s shoulders.

- Mere Access to atoms and rules is Necessary but not sufficient to creating new life forms.
- Lots of curves can fill this space, but only some will prove fruitful.
- The Combinatorial explosion within the design space requires a freedom for many to experiment.
- We need Collective wisdom and imagination.

# OpenMoko 2007 Software Stack.

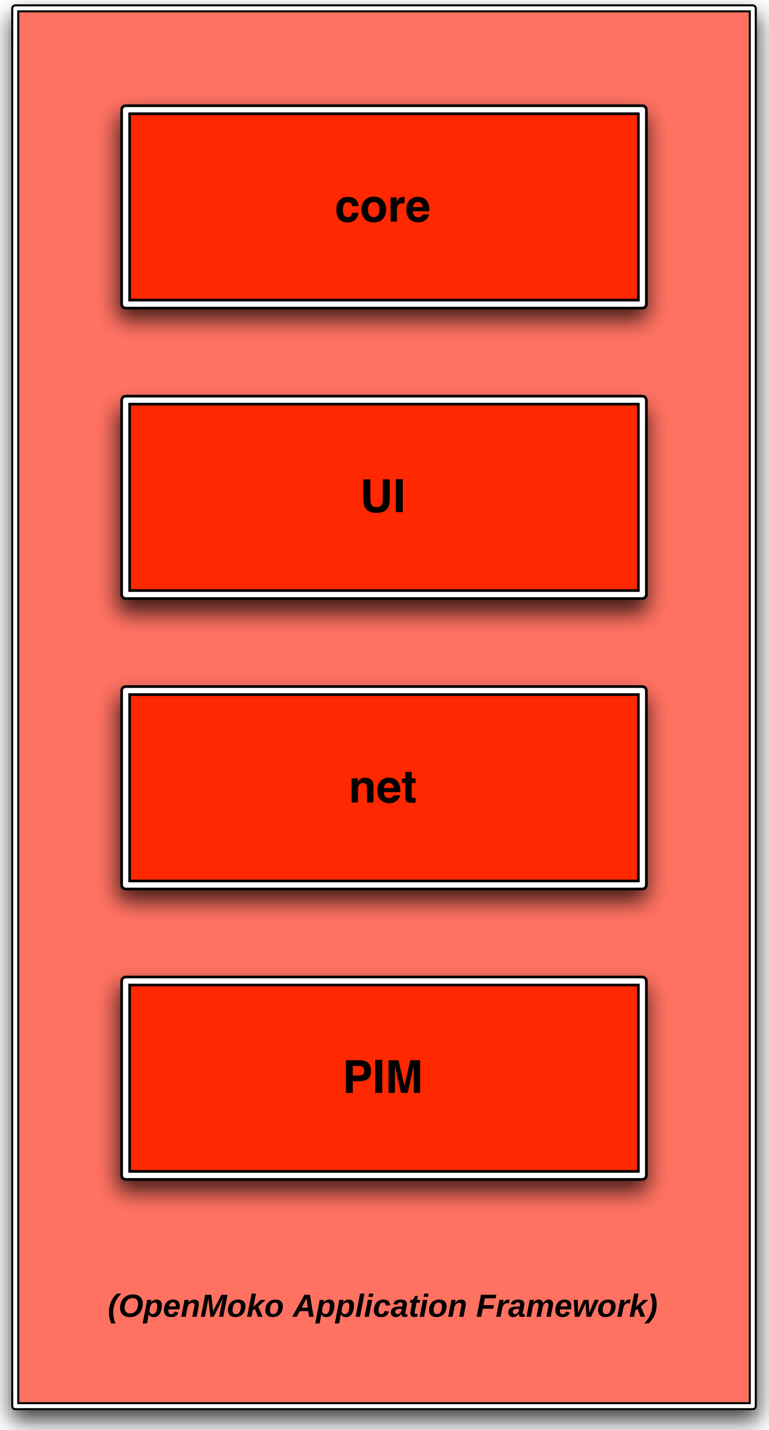


# 1) Atomic Access.

These are the building blocks of our system.

# OpenMoko's Application Framework.

- *libmokocore* – IPC, Device Control, Application State.
- *libmokoui* – Common look & feel.
- *libmokonet* – high-level connection queries.
- *libmokopim* – high-level PIM APIs.

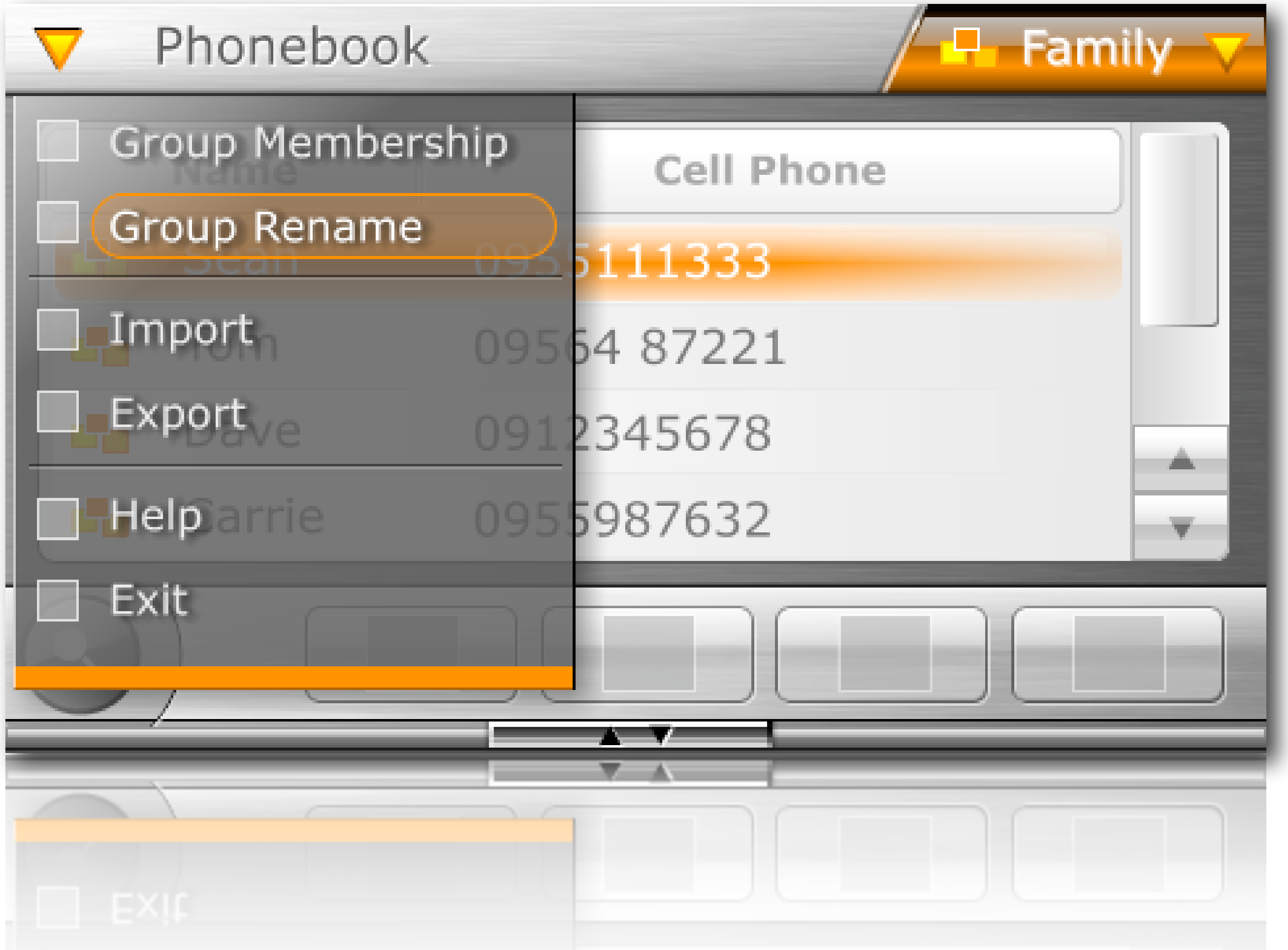


# libmokocore: At a Glance.

- OpenMoko IPC API
  - run_contacts_application
    (“new_phone_number”, “555-273-172”);
- Device Control API
  - device_set_display_brightness(device, 100);
  - s = device_get_signal_strength(device,
    MC_PERIPHERAL_GSM);
  - Uses dbus(-glib), libgconf, libgconf-bridge

# libmokoui: At a Glance.

- Full base GTK+ widgets
- Additional phone widget classes on top of GTK+



# libmokonet: At a Glance.

- peers = get_file_sinks( BT | INTERNET );
- at_home = gps_within_region( “at_home” );
- gsmconn = gsm_connection_new (“555-728-1829”);

# libmokopim: At a Glance.

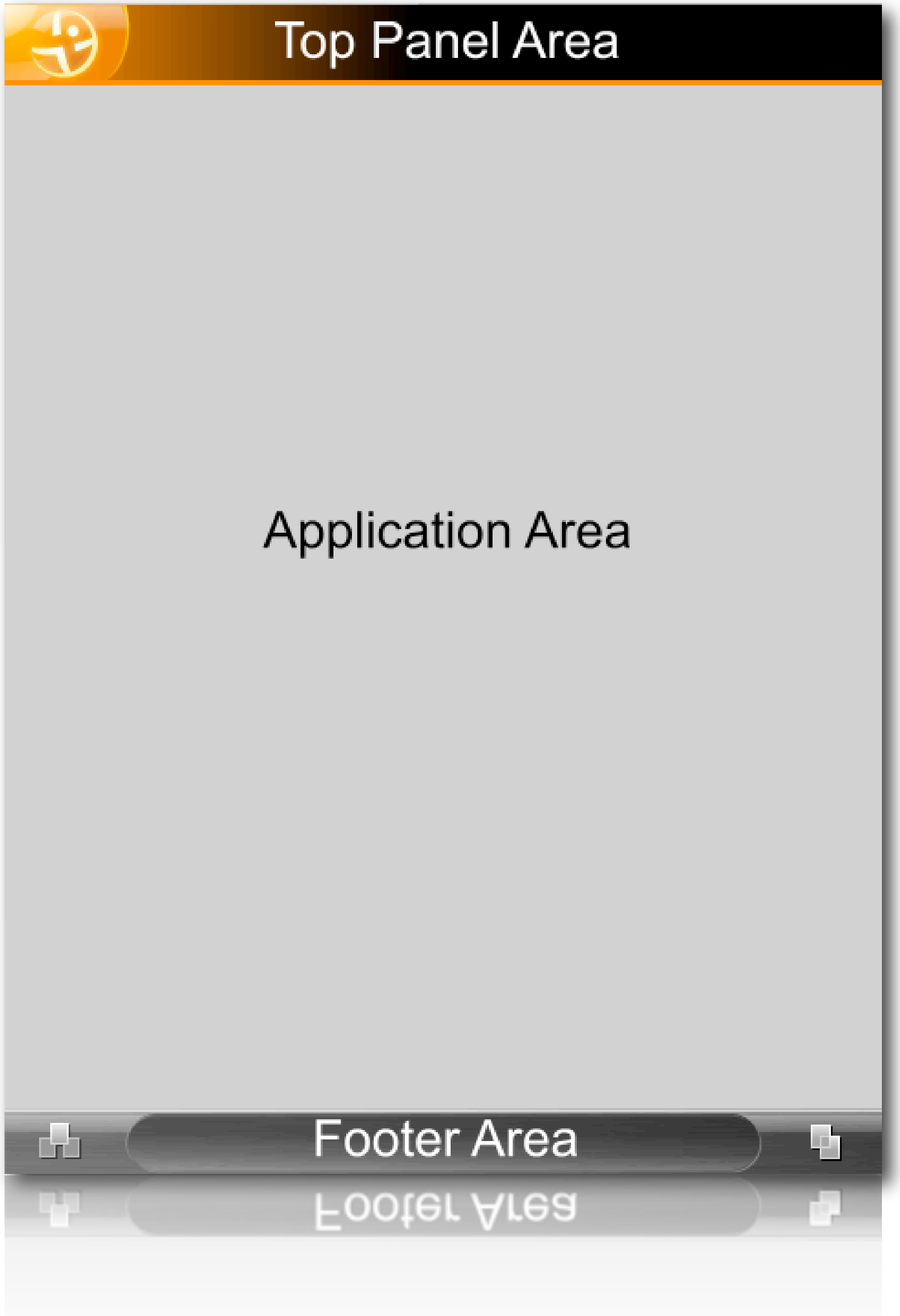
- Will probably never be written…
- Just use libebook, libecal, libcamel, and friends…

# 2) Freedom to Write Rules.

The ability to create your own combinations.

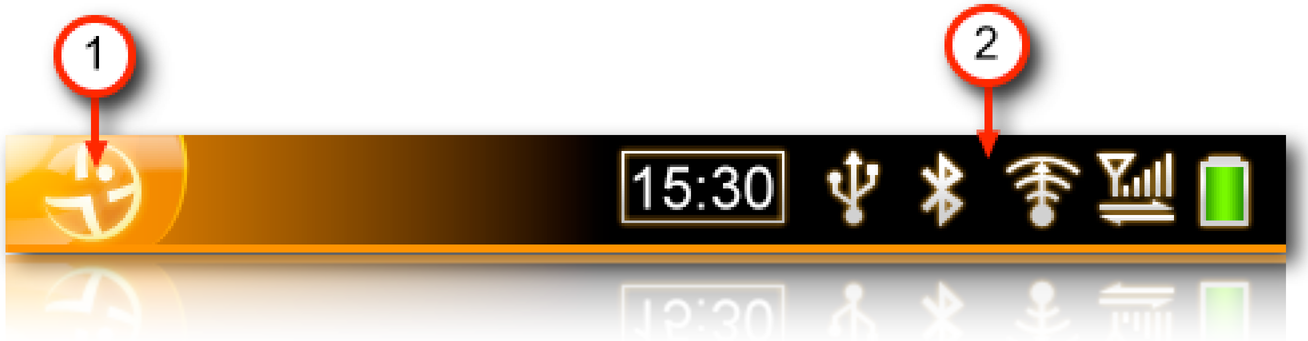
# The OpenMoko User Interface.

- openmoko-panel
- openmoko-<application>
- openmoko-footer



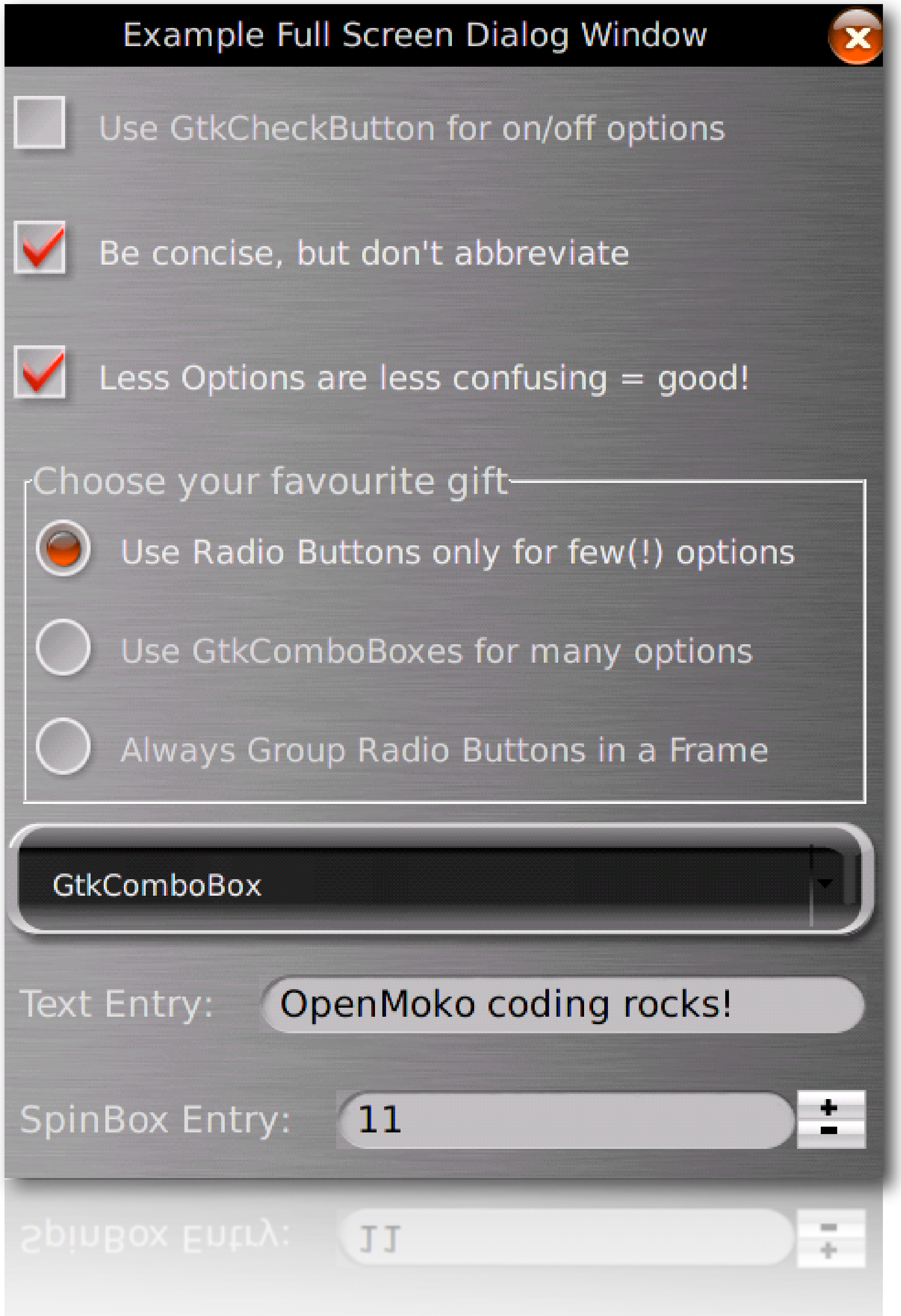
# openmoko-panel: At a Glance.

- Always visible and global for all applications.
- We just use matchbox-panel-2, lightweight gtk+-based panel
- Panel applet plugin host
  - Panel plugins are shared libraries
  - ${libdir}/matchbox-panel/*.so
  - Read on startup of mb-panel-2



# openmoko-<application>: At a Glance.

- Stylus applications
- Finger applications
- X11 legacy applications



Example Full Screen Dialog Window

□ Use GtkCheckButton for on/off options

☑ Be concise, but don't abbreviate

☑ Less Options are less confusing = good!

Choose your favourite gift

● Use Radio Buttons only for few(!) options

○ Use GtkComboBoxes for many options

○ Always Group Radio Buttons in a Frame

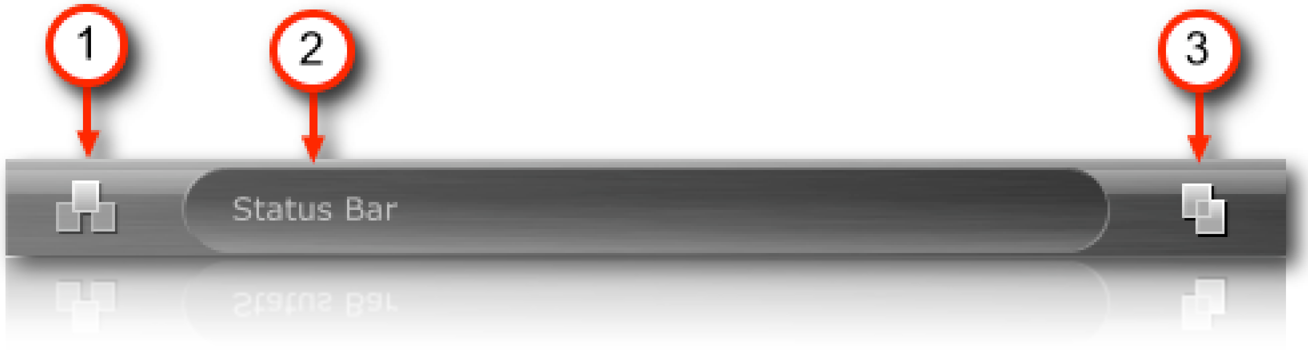
GtkComboBox

Text Entry: OpenMoko coding rocks!

SpinBox Entry: 11

# openmoko-footer: At a Glance.

- Task Manager
- Status Bar
  - Temporary Notification area
- Application Toggling



# 3) Labs to Experiment.

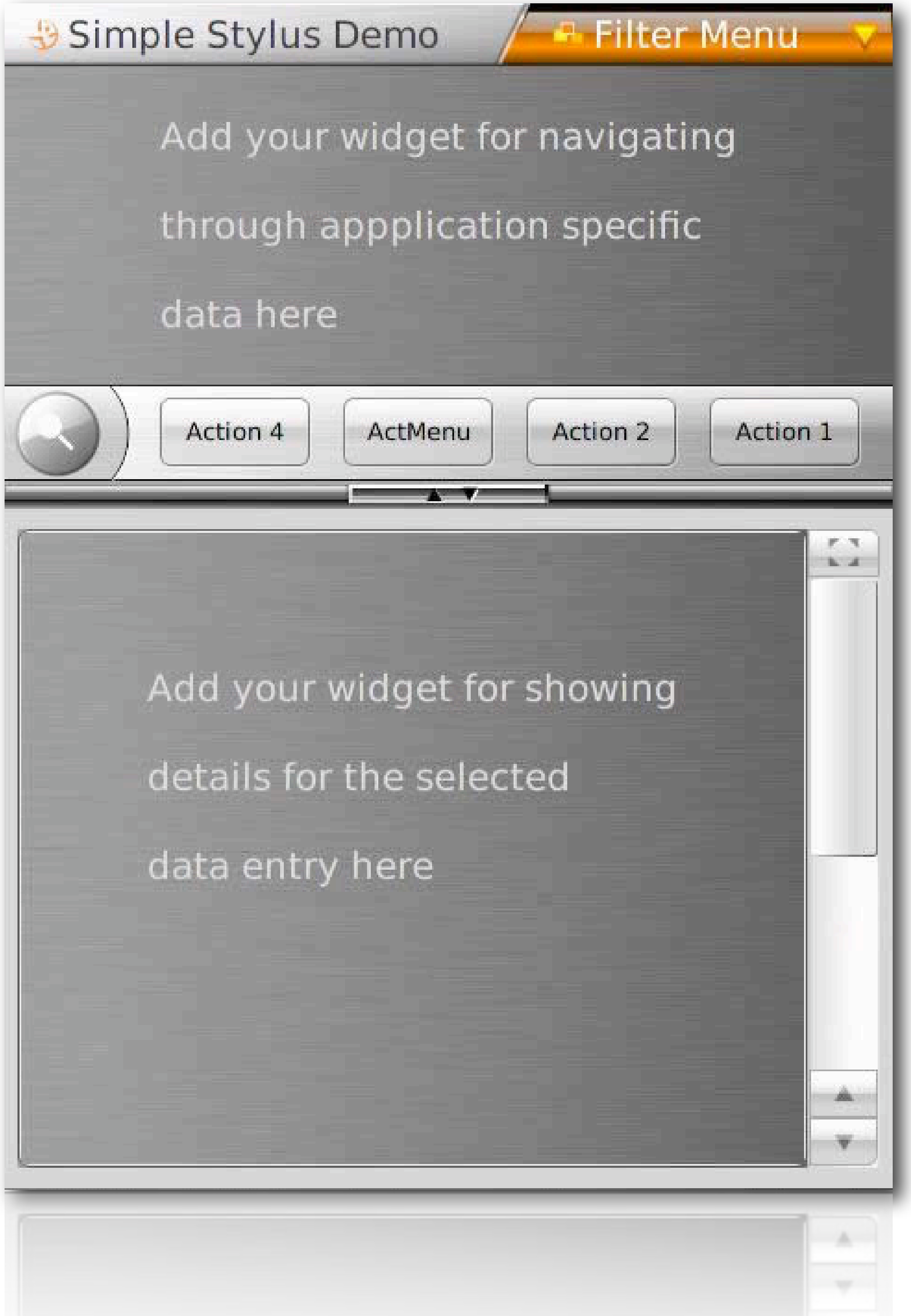
Lots of people trying new stuff.

# OpenMoko Application Development.

- Writing a Stylus Application
- Writing a Finger Application
- Using Other Widgets

# Stylus Applications: Overview.

- *MokoPanedWindow* – base class for stylus windows
- *MokoMenuBox* – application menu, filter menu
- *<Navigation Widget>* – e.g. GtkTreeView
- *MokoToolBox* – search, action buttons
- *<Details Widget>* – e.g. GtkLabel

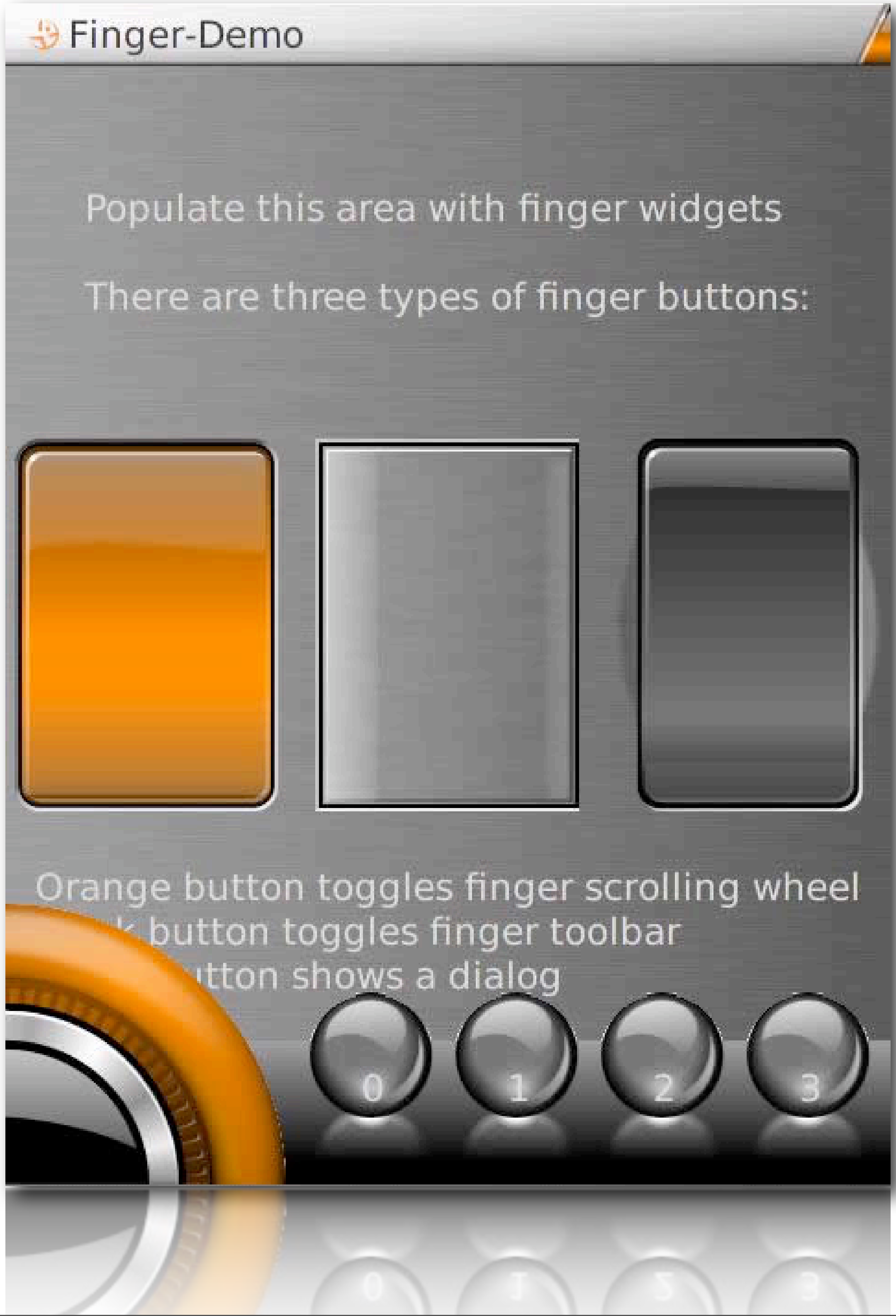


Add your widget for navigating through appplication specific data here

Add your widget for showing details for the selected data entry here

# Finger Applications: Overview.

- *MokoFingerWindow* – Base class for finger windows
- *MokoFingerButton* – Large, finger-friendly button
- *MokoFingerWheel* – Scrolling, mode changing (icon indicates mode)
- *MokoFingerToolBox* – Three (or more) tools per page, multiple pages possible



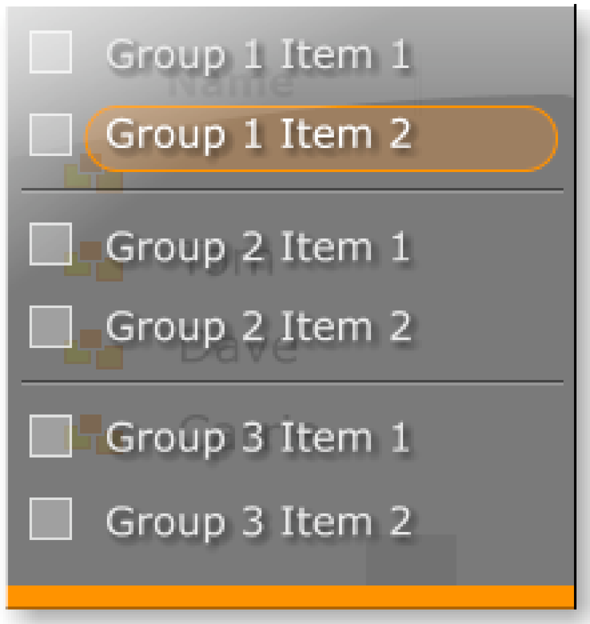
Populate this area with finger widgets

There are three types of finger buttons:

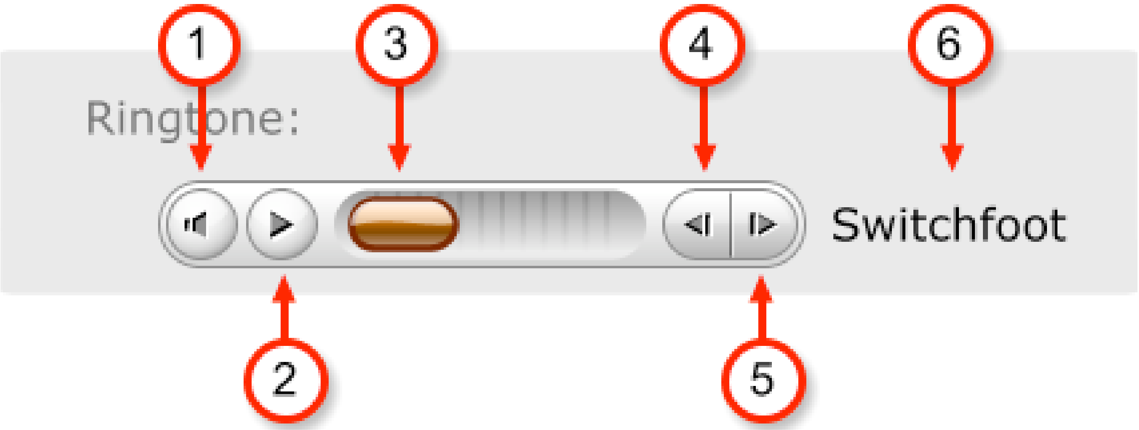
Orange button toggles finger scrolling wheel
k button toggles finger toolbar
tton shows a dialog

# Other Widgets.

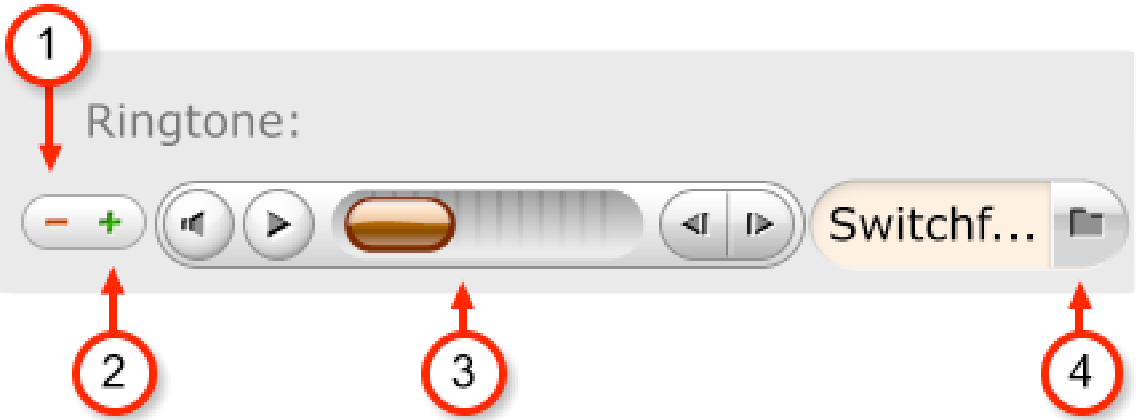
- *MokoDialogWindow* – Full-screen modal dialog, can use any Gtk+ widget
- Field Widgets
- View mode
- Edit mode
- More...



□ Group 1 Item 1
□ Group 1 Item 2
□ Group 2 Item 1
□ Group 2 Item 2
□ Group 3 Item 1
□ Group 3 Item 2



Ringtone: Switchfoot



Ringtone: Switchf...

# 4) Feedback.

Collective experimentation leads to new life forms.

# 2007 Finger Applications.

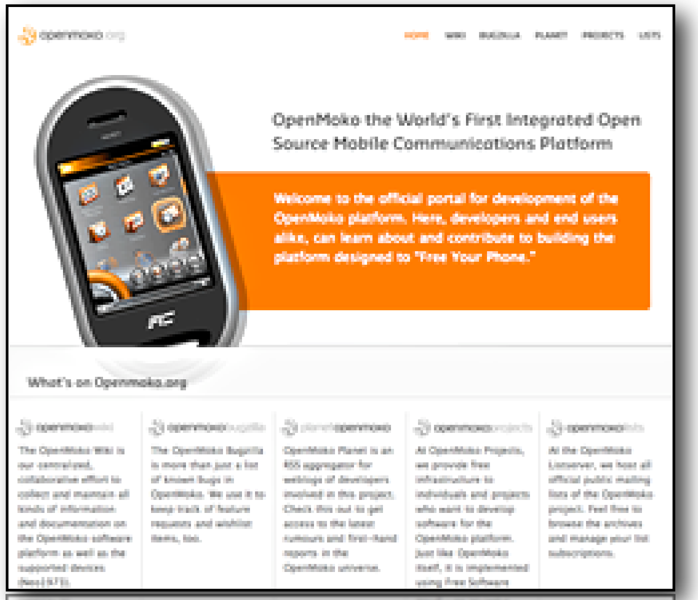
| PHASE 1 | PHASE 2 |
|---|---|
| Dialer | Clocks<br>Screen Saver<br>Calculator<br>Unit Converter<br>Game<br>Guitar Tuning<br>Code Memo<br>*Your Applications...* |
| Main Menu | |
| Music Player | |
| History | |

# 2007 Stylus Applications.

| PHASE 1 | PHASE 2 |
|---|---|
| Contacts | Feed Reader<br>Messages<br>Preferences<br>Media Player<br>Sketchbook<br>Terminal<br>IM<br>Web Browser<br>Reader<br>System Info<br>*Your Applications...* |
| Dates | |
| Application Manager | |
| Today | |

# Community Resources.

{openmoko.org}



openmoko.org

HOME WIKI BUGZILLA PLANET PROJECTS LISTS

OpenMoko the World's First Integrated Open Source Mobile Communications Platform

Welcome to the official portal for development of the OpenMoko platform. Here, developers and end users alike, can learn about and contribute to building the platform designed to "Free Your Phone."

What's on Openmoko.org

openmokowiki

The OpenMoko Wiki is our centralized, collaborative effort to collect and maintain all kinds of information and documentation on the OpenMoko software platform as well as the supported devices (Neo1973).

openmokobugzilla

The OpenMoko Bugzilla is more than just a list of known bugs in OpenMoko. We use it to keep track of feature requests and wishlist items, too.

planetopenmoko

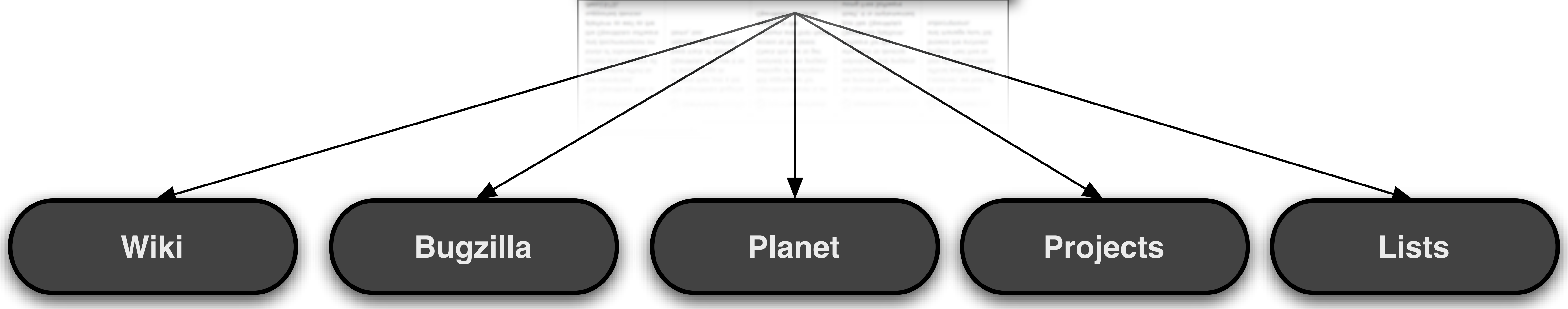
OpenMoko Planet is an RSS aggregator for weblogs of developers involved in this project. Check this out to get access to the latest rumours and first-hand reports in the OpenMoko universe.

openmokoprojects

At OpenMoko Projects, we provide free infrastructure to individuals and projects who want to develop software for the OpenMoko platform. Just like OpenMoko itself, it is implemented using Free Software

openmokolists

At the OpenMoko Listserver, we host all official public mailing lists of the OpenMoko project. Feel free to browse the archives and manage your list subscriptions.



Wiki

Bugzilla

Planet

Projects

Lists

In 1973, Marty Cooper invented the mobile phone. This gave birth to an industry. We’re going to revolutionize it again. Only this time, you will write the rules.

Welcome to the New 1973. The future is open.

# The Neo1973: Write Your Own Rules.

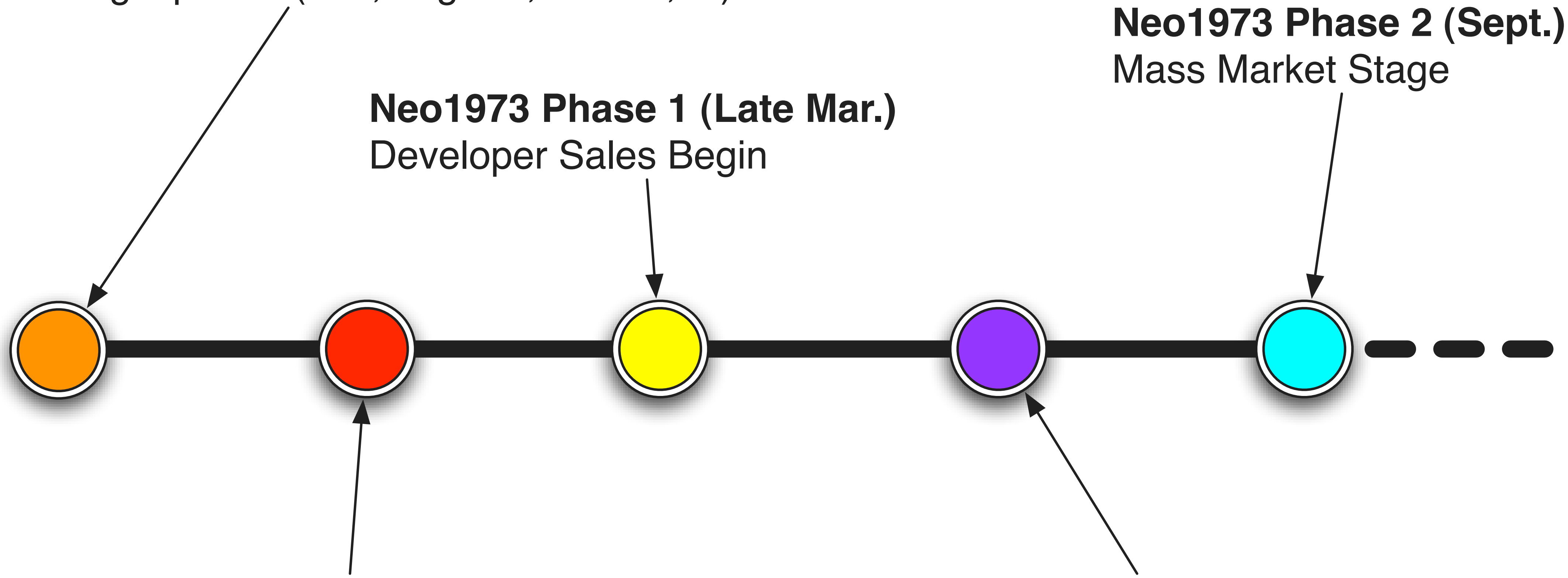
# Your Mobile Lab for Experimentation.

# Create New Building Blocks.

# Cost Breakdown.

| | Description | Retail |
|---|---|---|
| Standard Kit | Neo1973<br>Battery<br>Headset<br>Compact Charger<br>Carrying Case<br>Stylus<br>Lanyard<br>MicroSD Card<br>Micro USB Connectivity Cable<br>Instruction Manual and Warranty | US$350 |
| Car Kit | Windshield Mount and Device Holder<br>Car Charger<br>External Antenna | US$75 |
| Hacker’s Lunchbox | Development Board<br>Battery<br>Compact Charger for Development Board<br>FPC<br>Shoulder Strap<br>USB A-B | US$200 |

# Our 2007 Roadmap.



**Neo1973 Open R&D (Feb. 12)**
openmoko.org Opened (wiki, bugzilla, source, ...)

**Neo1973 Phase 0 (Early Mar. )**
First Phones are Freed

**Neo1973 Phase 1 (Late Mar.)**
Developer Sales Begin

**Neo1973 Phase 1+ (Jun.)**
Hardware Refresh

**Neo1973 Phase 2 (Sept.)**
Mass Market Stage

“Never send a human to do a machine’s job.”

Agent Smith, 1999.

# Why in God’s name don’t we use phones and humans to do this...

- Schedule a call on your calendar
- Get your approval, check your time zone.
- Request to dial you at the appointed time...

**“Neo... Call Mickey when he and I are both available.”**

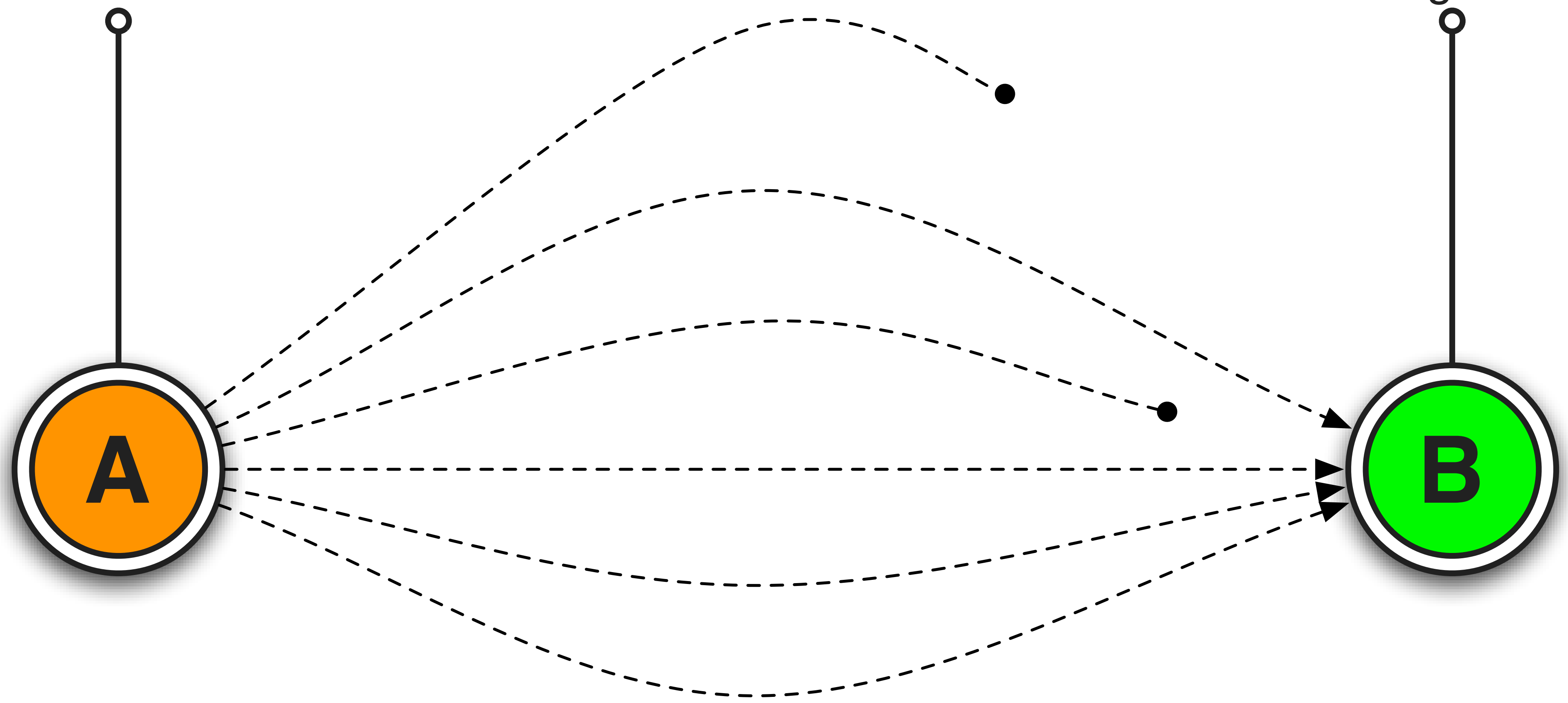
# The PC is maladaptive. The Phone is maladaptive. Don’t follow the phone. Leapfrog it. The key is to achieve what the PC and the phone *intended*.

Computing everywhere. Intuitive computing. Computing that is as natural to us as finger painting.

# The 21st Century's Opportunity.

**OpenMoko**
Freedom to write new
rules of combination.

**Ubiquitous Computing**
Our devices learn us rather than
us learning our devices.



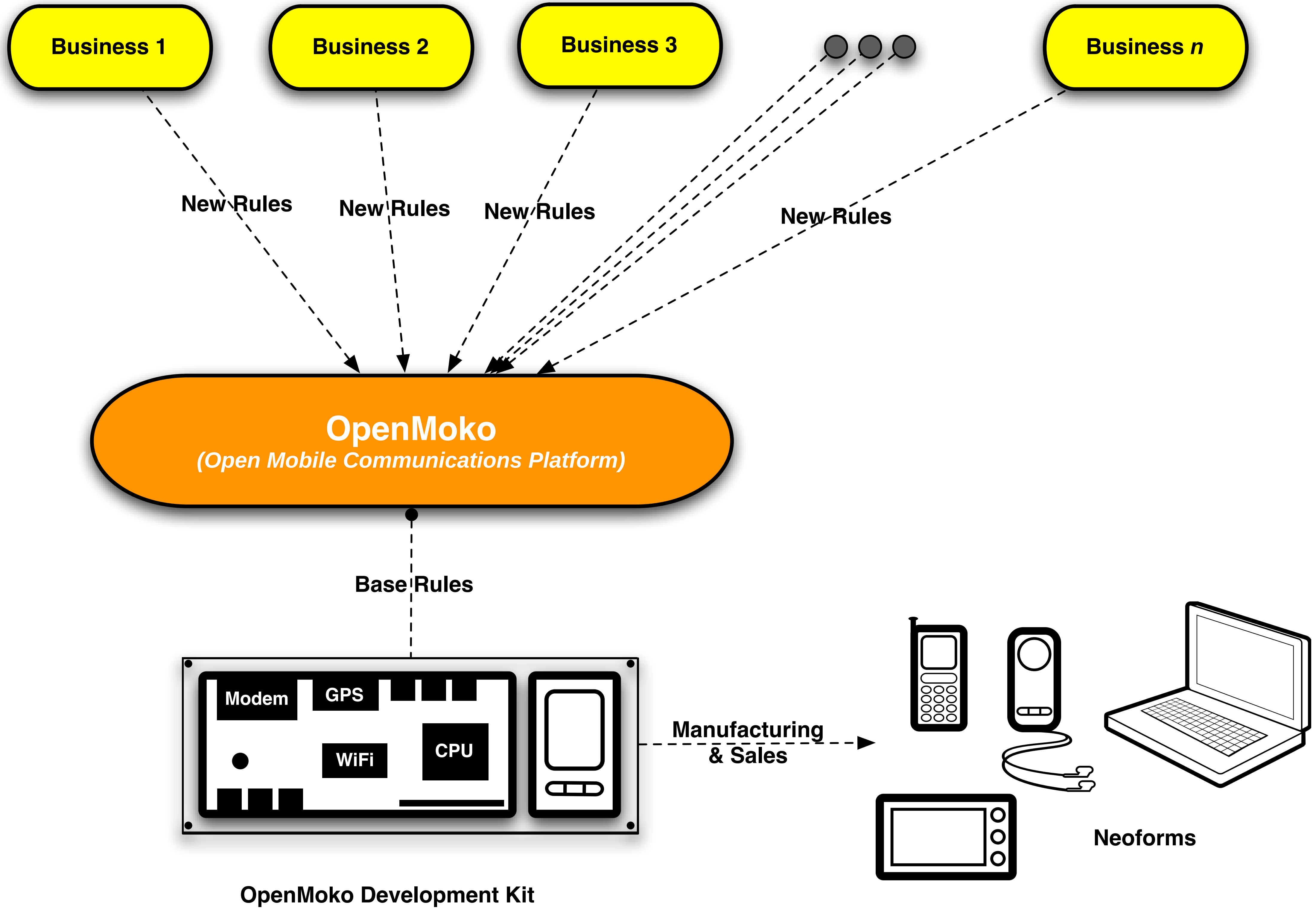
**{Simple Systems}**

**{Complex Forms}**

# How do simple systems evolve into complex forms?

- Open access to Essential building blocks
  - Processor, input subsystems, output subsystems
- Open access to Rules for combining and controlling these subsystems
- Freedom by many to experiment
- A marketplace to reward Success
- OpenMoko provides this stuff

# Our Business Model.



# Now, “Free Your Phone.”

Thanks for Your Time.
Mickey Lauer & Sean Moss-Pultz