Maemo 5 с точки зрения пользователя

Все, что касается изменений, сразу заметных пользователю: увеличение размера элементов пользовательского интерфейса, отсутствие “несветящихся” элементов (если элемент неактивен - сразу исчезает), удаление фокуса, максимальное число элементов - 10, меню больше не иерархическое, стековое меню и др.
http://wiki.maemo.org/Documentation/Maemo_5_Developer_Guide/Porting_Software/Redesigning_From_Maemo_4_to_Maemo_5

Maemo 5 с точки зрения разработчика

Рассматривается интерфейс и модули определения географического положения

Основные компоненты графического интерфейса пользователя

Сюда входят: C library, Xlib, Glib, GDK, GDK, Pango, ATK, GTK+, Hildon. Коротко о них.
http://wiki.maemo.org/Legacy_Maemo_5_Documentation/Graphical_UI_Tutorial/Introduction

Hildon - изменения в Maemo5

На основе таблицы
http://repository.maemo.org/stable/fremantle/4.1.2_vs_5.0_content_comparison.html

GTK - изменения в Maemo5

На основе таблицы
http://repository.maemo.org/stable/fremantle/4.1.2_vs_5.0_content_comparison.html

Qt

http://doc.trolltech.com/4.5/index.html

Пакет qt4-x11

http://packages.debian.org/ru/source/lenny/qt4-x11

Компиляция Qt приложений

http://wiki.maemo.org/Qt4Hildon#Overriding_the_Qt_Maemo_changes

Портирование QT приложений на Maemo5

http://wiki.maemo.org/Qt4Hildon#Overriding_the_Qt_Maemo_changes
http://wiki.maemo.org/Packaging_a_Qt_application

Сосуществование Qt и GTK+ на Maemo5

Новые Возможности написания интерфейса, предоставленные появлением Qt

Использование модулей для определения географического положения

Location framework в MAEMO4: liblocation

Liblocation — библиотека, состоящая из нескольких модулей и предоставляющая разработчику приложений для Maemo средства определения географического положения. Liblocation включает в себя модули: LocationGPSDevice, LocationGPSDControl, location-distance-utils, location-misc. Модуль LocationGPSDevice содержит набор типов данных и функций, позволяющих принимать информацию от демона местоположения (например, GYPSY), который в свою очередь взаимодействует с GPS устройством. Модуль LocationGPSDControl определяет объект, который позволяет управлять соединением с демоном местоположения. В модуле определены функции, информирующие демона об открытии или закрытии соединения с ним. Демон положения взаимодействует с устройством до тех пор, пока хотя бы одно приложение поддерживает с этим демоном соединение. Модуль location-distance-utils предоставляет разработчику средство определения расстояния между двумя точками на поверхности Земли. Модуль location-misc содержит набор дополнительных функций для работы с liblocation. На данный момент определяет одну функцию, которая осталась для совместимости и не рекомендуется для использования при разработке новых программ. Maemo Diablo включает liblocation версии 0.30. Реализации версий liblocation до 0.30 были основаны на взаимодействии с демоном местоположения GPSD. Следующая версия liblocation 0.90 основана на GYPSY и D-Bus.

Location framework в MAEMO5: gypsy vs gpsd

Maemo Diablo для работы с GPS устройствами использовало демон GPSD. Maemo Fremantle содержит liblication версии 0.99, которая вместо GPSD использует GYPSY. GYPSY был написан, чтобы исправить проколы, обнаруженные в GPSD. Одним и з проколов была проблема

выделения памяти (allocating memory). Разработчики GPSD категорически не рекомендовали использовать функции malloc/free ((http://gpsd.berlios.de/hacking.html#malloc). Кроме того, в GPSD клиенту необходимо открывать сокет для взаимодействия с сервером GPSD. Обмен информацией достаточно сложный. В GYPSY взаимодействие основано на использовании мощной сигнальной системы D-Bus. Существует ряд других причин, по которым предпочтительнее использовать GYPSY, а не GPSD.

Библиотеки нижнего уровня для определения геграфического положения в Maemo

Библиотеки ниженего уровня в Maemo 4.x (Diablo)

К библиотекам нижнего уровня для определения положения в Maemo Diablo относятся библиотеки: libgpsbt, libgpsmgr, gpsd daemon.

GPS BT (libgpsbt)

Application that needs GPS data can use this API to manage GPS daemon start and stop and to find out what serial port device (/dev/rfcommX) the GPS device is using. The API uses services provided by libgpsmgr that handles the GPS daemon start and stop.

GPS Manager (libgpsmgr)

Application that needs GPS data can use this API to manage GPS daemon start and stop. This is needed in small handhelds where it is not good to have gpsd running all the time (because of battery and memory constraints) and to guarantee that only one gpsd process is running simultaneously even if multiple processes need the GPS data. The idea of the API is to allow multiple applications to use gpsd services but only when there is at least one application that needs GPS data. If no application is running, the gpsd is not started at all or gpsd stops itself automatically (to save memory and battery life).

Библиотеки нижнего уровня в Maemo 5.0 (Fremantle)

В MAEMO5 отсутствуют библиотеки нижнего уровня libgbsbt, libgpsmgr, gpsd daemon.Вместо этих библиотек в MAEMO5 предполагается использование liblocation либо API для обращения к демонам, например к GYPSY.

Пример работы с liblocation в Maemo.

Ниже приведен пример работы с liblocation в Maemo 5, который был взят с сайта maemo.

Here is a complete standalone example using liblocation. It starts location services after program is started, then when first fix arrives, prints it, stops services, and shutdowns. #include

<location/location-gps-device.h> #include <location/location-gpsd-control.h>

static void on_error(LocationGPSDControl *control, LocationGPSDControlError error, gpointer data) { g_debug("location error: %d... quitting", error); g_main_loop_quit((GMainLoop *) data); }

static void on_changed(LocationGPSDevice *device, gpointer data) { if (!device) return;

if (device→fix) { if (device→fix→fields & LOCATION_GPS_DEVICE_LATLONG_SET) { g_debug("lat = %f, long = %f", device→fix→latitude, device→fix→longitude); location_gpsd_control_stop((LocationGPSDControl *) data); } } }

static void on_stop(LocationGPSDControl *control, gpointer data) { g_debug("quitting"); g_main_loop_quit((GMainLoop *) data); }

static gboolean start_location(gpointer data) { location_gpsd_control_start((LocationGPSDControl *) data); return FALSE; }

int main(int argc, char *argv[]) { LocationGPSDControl *control; LocationGPSDevice *device; GMainLoop *loop;

g_type_init();

loop = g_main_loop_new(NULL, FALSE);

control = location_gpsd_control_get_default(); device = g_object_new(LOCATION_TYPE_GPS_DEVICE, NULL);

g_object_set(G_OBJECT(control), "preferred-method", LOCATION_METHOD_USER_SELECTED, "preferred-interval", LOCATION_INTERVAL_DEFAULT, NULL);

g_signal_connect(control, "error-verbose", G_CALLBACK(on_error), loop); g_signal_connect(device, "changed", G_CALLBACK(on_changed), control); g_signal_connect(control, "gpsd-stopped", G_CALLBACK(on_stop), loop);

g_idle_add(start_location, control);

g_main_loop_run(loop);

g_object_unref(device); g_object_unref(control);

return 0; }

You can compile this example with following command: gcc -Wall `pkg-config –cflags glib-2.0, liblocation –libs glib-2.0, liblocation` -o test test.c

From:
http://wiki.osll.ru/ - **Open Source & Linux Lab**

Permanent link:
http://wiki.osll.ru/doku.php/etc:common_activities:maemo:maemo4_maemo5?rev=1256914973

Last update: **2009/10/30 18:02**