Changes to positmobile codebase

Packages

The following packages has been added to the existing list of positmobile packages:

⇒ Package `org.hfoss.posit.android.api:`

This package contains all interface declarations specifying functionality each plugin component must implement.

Classes and interfaces:

• `FindPluginManager.java`: main class responsible for loading plugin components. Singleton instance of this class, once initialized, contains generic references to each of the plugin components, such as `FindFactory` and `FindDataManager` static singleton instances, as well as `FindActivity` and `ListFindsActivity` Class-type objects for a currently loaded plugin (see `res/raw/plugin_preferences.xml` description under Configuration files section).

• `Find.java`: an abstract class enforcing constructors which must be provided by extending subclass.

• `FindInterface.java`: interface implemented by all `Find` objects. This interface enforces list of public methods which must be implemented by extending subclasses.

• `FindActivity.java`: pure abstract class extending `Activity` class and implementing `OnClickListener`, `OnItemClickListener`, and `LocationListener` interfaces, all "pure virtual" functions of which must be implemented by extending find activity subclasses.

• `ListFindsActivity.java`: pure abstract class extending `ListActivity`. Must be extended by plugin-specific activities wishing to display list of finds.

• `FindFactory.java`: abstract singleton factory class implementing `FindProviderInterface`. The aim of subclasses extending this class is to create instances of plugin-specific Find objects. By using generic references to plugin-specific Find factories, rest of POSIT code can request factories to create finds without the need to know specific type of find that the plugin implements.

• `FindProviderInterface.java`: interface mimicking constructor signatures of `FindInterface` objects. Factory implementing this interface uses its method parameters to create new `Find` objects.

• `FindProvider.java`: convenience non-instantiable class wrapping sequence of chained calls to `FindFactory` to create a new `Find` object.

• `FindActivityProvider.java`: convenience non-instantiable class wrapping sequence of chained calls to `FindPlugManager` to get a reference to Class-type objects of a `FindActivity` and `ListFindsActivity` subclasses.

• `FindDataManager.java`: abstract class implementing `FindDataManagerInterface`, enforcing singleton functionality of plugin-specific find data managers.

• `FindDataManagerInterface.java`: interface enforcing ability of plugin-specific find data managers to load find data (for example, a bitmap image) from `Uri` and store it as base64 string, which later could be used for transmission over to the server. Reverse functionality, involving decoding of base64 string and saving encoded data as `Uri` must also be implemented.

⇛ Package `org.hfoss.posit.android.photofind:`

This package contains what used to be positmobile's only way to display finds. The code is mostly identical to that which was once contained in `Find.java` and `FindActivity.java`. Some refactoring has been applied to make this code comply with plugin specification, i.e. extending necessary superclasses and following new calling conventions. Additional classes include:

• `PhotoFindFactory.java`: implementation of `FindFactory` which creates finds specific to find activity which displays a photograph.

• `PhotoFindDataManager.java`: implementation of `FindDataManager` specializing in encoding and decoding bitmaps (find photographs) to and from base64 strings.

⇛ Package `org.hfoss.posit.android.photofindsorted:`

Package identical to `org.hfoss.posit.android.photofind` with exception of containing Eran Henig's change to make list of finds sortable either by date or by find's name.

⇛ Package `org.hfoss.posit.android.provider:`

• `PositDbHelper.java`: an individual instance of this class, meaning, individual instance of database connection used to be created for every find that was downlaoded, created, or modified. This class was converted to singleton to reduce overhead.

⇛ Other packages

To conform with presence of `FindPluginManager` and its services, many references to specific classes and activities have been replaced with calls to `FindPluginManager` which in turn decides which plugin-specific classes and activities shall provide said services.

Configuration files

The following xml scripts have been added and/or modified:

• AndroidManifest.xml: changes in activity names are reflected here. Each plugin must declare it's activities here so that they may be loaded by plugin manager.

• `res/raw/plugin_preferences.xml`: main manifest file describing which plugins are installed and which one is currently active. Each <Plugin ... /> element indicates whether or not it is active, and specifies names of factory and activity classes which implement plugin api. These classes are then dynamically loaded and internally stored by `FindPluginManager`, which can then provide references to plugin components on request from rest of POSIT code.

Pending issues

The following issues remain unresolved:

• Photograph-specific database table and column names on both client and positweb server (See `PositDbHelper.java`). In order to reflect arbitrary type of find data, table names on both mobile and server databases need to be changed to reflect that. In addition, it must be ensured that synchronization works with arbitrary find data.

• Photograph-specific HTTP header content generated by `Communicator` class when querying server for find data. Changing this will require modifications to both client- and server-side code.

• Adding plugins api to the positweb web interface. Requires modifications to server-side code.