Developing the Mobile Solution

Requirements:

- The mobile solution needs to have the mobile solution type
- If the data shown in mobile app is user specific the mustAuthenticate property has to be enabled
- The mobile plugin has to be installed

Once the Solution Type Is Set to Mobile the:

- solution explorer view will filter to show only the capabilites possible in mobile. (API wise)
- form designer/palette will change to place mobile interface elements utilizing a different layout manager.
- each form shown in the mobile client without passing a (related) foundset (like first form) will need a offlineDataDescription.addFoundSet call in the mobile service solution

Limitations:

- A records dataproviders should be accessed through the record itself, which means first a record object needs to be retrieved before the dataproviders from such a record should be accessed.
  Example:

  ```
  var rec = foundset.getSelectedRecord();
  if (rec.phone_direct) plugins.mobile.call(rec.phone_direct);
  ```

  So try to avoid direct access of those variables, so use 'record.column' instead of just 'column' for the selected record in a form method.
- new records pks always become UUID's values, while UUID are not required for pks for any data delivered by the server, the service solution would need to translate UUID's to local ids upon retrieval of new records.
  When using "push" from the mobile plugin, the service solution must return, for the newly created records, the server side pk; that means, the ws_create must return the pk as an object with uuid as key, pk as value, or if a bulk update is used in the service solution, ws_update must return the object that performSync returns, that is an object with all uuid,pk mapping returned by ws_create methods;
- only single pk are supported on entities. (no compound pk support)
- no SQL support, everything has to happen via relations/foundset navigation from the provided root foundset by the service solution.
- only custom valuelists are supported, but application.setValueListItems is present
- Be careful with the Form variables or Global variables names, prefix or post fix them with an nice value so that you don't have global variables named 'status' or 'document' these names must be avoided because they collide with the window.xxx object of the browser.

Within these limitations at form design and API level all business logic should work.

Initialize:

A mobile client works offline, so the first time it must do a synchronization with the server, this will be automatically called when the client starts up.It will then also ask for credentials first if the solution requires authentications.

With Servoy 7.4 this automatic initial synchronization can be avoided if the first form doesn't have a datasource, so it doesn't need data. This way you can control the sync and login completely in your solution. Be aware no initial login will be shown either, if you want to make sure a login form is always presented you have to make your own (without datasource) and show this as first form, calling security.authenticate with the result for future use.

Assumptions:

Servoy mobile assumes the same user is using the same or his own device.
If this is not the case, i.e. any user can use any device, you will have to make sure your own login form is present and shown each time the application opens.This is possible by making it the first form (without datasource), you might also want to clear the local storage to make sure no one sees other people data.

Supported plugins:

- Mobile plugin (especially designed for using mobile device capabilities, like GPS and making calls)
- Dialog plugin showWarningDialog function

Related Pages:

- Using External CSS
- Using External JavaScript

- Using JQueryMobile JavaScript Library