

TFM.co Mobile Market Manager

User Guide and Developer Documentation

By Alex Lindeman, Bill Brooks

Last updated April 26th, 2015

Table of contents

[User Guide](#)

[Managing a market day](#)

[Starting a market day](#)

[Creating a new market day](#)

[Reopening an old market day](#)

[Editing market day information during a market day](#)

[Recording transactions with customers](#)

[Recording redemptions with vendors](#)

[Reconciling token totals](#)

[Reconciling terminal totals](#)

[Closing the market day](#)

[Managing vendors, staff, and locations](#)

[Reports](#)

[Managing reports](#)

[Raw data reports & database dumps](#)

[Managing data through iTunes](#)

[Adding data to the device](#)

[Taking data off the device](#)

[Importing data](#)

[Import format](#)

[Vendors](#)

[Staff](#)

[Locations](#)

[Erase data](#)

[All market days](#)

[All data, except reports](#)

[All data](#)

[Using the console](#)

[Developer Documentation](#)

[Project settings](#)

[Project file structure](#)

[Enumerated value reference](#)

[Types used in reports](#)

[Ethnicity \(Transactions.cust_ethnicity\)](#)

[Frequency \(Transactions.cust_frequency\)](#)

[Gender \(Transactions.cust_gender\)](#)

[Position \(MarketStaff.position\)](#)

[Internal types](#)

[ImportType](#)

[ImportFormat](#)

[ImportDumpOptions](#)

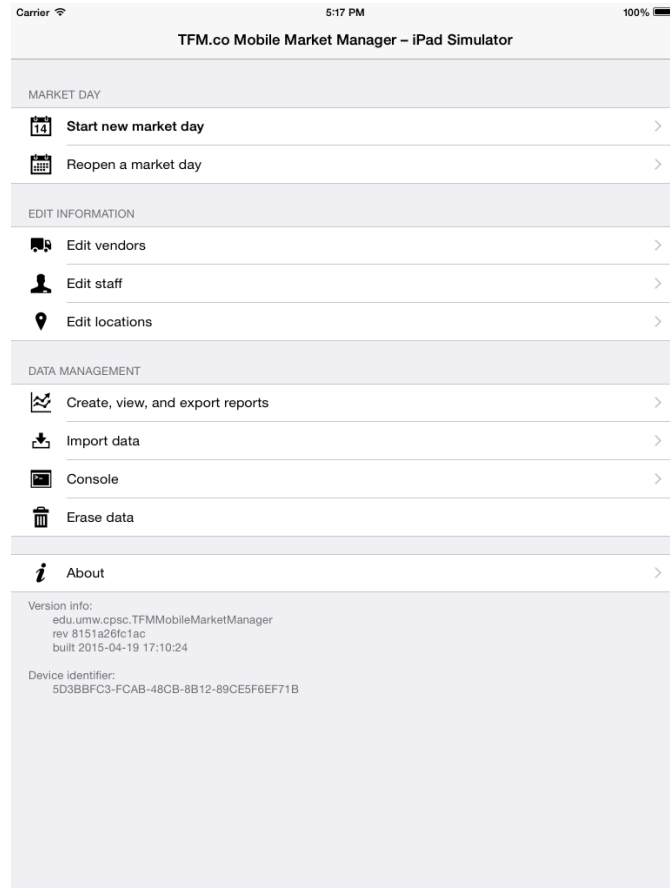
[To-do list](#)

[User interface](#)

[Core functionality](#)

[Low-priority](#)

User Guide



The Market Manager main menu

Managing a market day

Starting a market day

Creating a new market day

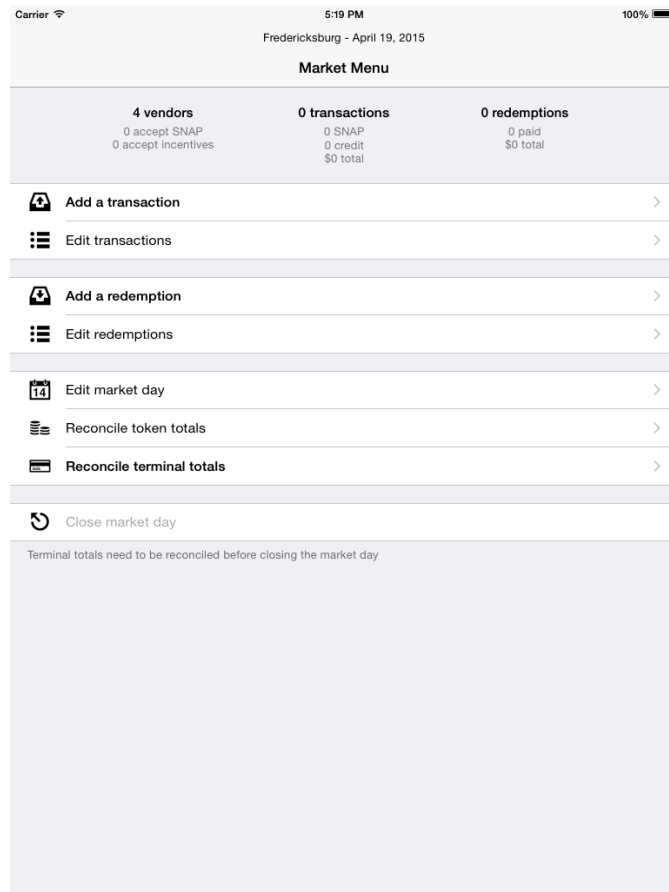
The New Market Day window

From main menu, press *Start a new market day*. From here, select a location, vendors who will be present at the market day, a date (already set to today's date), a start and end time (already set to the current time), staff, and any notes. This information can be changed later. Press *Open* at the upper right to open the market day and begin recording transactions.

Reopening an old market day

From main menu, press *Reopen a market day*. Select a previously closed market day, press *Open* at the upper right, and it will take you to the Market open menu where transactions can be recorded. You can also create a new market day from this screen by pressing the + icon at the upper right.

Once the market day is open, the Market Open menu will appear, with the name of the current market day on top.



The Market Open menu

Editing market day information during a market day

From the Market Open menu, press *Edit market day*. This will display the same form used when opening the market day, and you can edit any of its information and press *Save* to save your changes.

Recording transactions with customers

From the Market Open menu, press *Add a transaction*, fill out the form, and press *Save* at the top (or *Close* to cancel).

To view past transactions, press *View transactions*. You cannot edit or delete transactions that have already been made, but you may mark the transaction as invalid (by swiping the transaction you'd like to mark as invalid to the left and pressing *Mark invalid*, or by tapping it and checking the *Mark as invalid* option at the bottom and saving), and record a new one instead. Invalid transactions will be crossed and greyed out, and won't factor into totals.

Recording redemptions with vendors

From the Market Open menu, press *Add a redemption*, fill out the form, and press *Save* at the top (or *Close* to cancel).

To view past redemptions, press *View redemptions*. You cannot edit or delete redemptions that have already been made, but you may mark the redemption as invalid (by swiping the

redemption you'd like to mark as invalid to the left and pressing *Mark invalid*, or by tapping it and checking the *Mark as invalid* option at the bottom and saving), and record a new one instead. Invalid redemptions will be crossed and greyed out, and won't factor into totals.

Reconciling token totals

The screenshot shows a mobile application interface for reconciling token totals. The screen has a title bar with 'Close' on the left and 'Save' on the right. The main content is divided into three sections, each with a header and a table of token types and counts.

DISBURSED TOKENS	
Credit tokens	0
SNAP tokens	0
Bonus tokens	0

REDEEMED TOKENS	
Credit tokens	0
SNAP tokens	0
Bonus tokens	0

DIFFERENCE	
Credit tokens	0
SNAP tokens	0
Bonus tokens	0

The token reconciliation process is optional before closing the market day.. The first section, *Disbursed Tokens*, contains the number of tokens that have been given out during the market day. The second section, *Redeemed Tokens*, contains the number of tokens that have been redeemed at the end of the market day. The *Difference* section at the bottom shows any discrepancy between the token counts.

Reconciling terminal totals

Close Reconcile Terminal Totals Verify

ON TERMINAL

SNAP	Credit	Total
Value (\$)	Value (\$)	Value (\$)
Transactions	Transactions	Transactions

ON THIS DEVICE

SNAP	Credit	Total
Value (\$)	0	Value (\$)
Transactions	0	Transactions
		0

If the totals do not match, it is usually an error involving:

- the terminal not processing a transaction
- a transaction incorrectly marked or not marked as invalid
- a typo either on this device or the terminal

The terminal total reconciliation process must be completed before closing the market day. The upper table contains the terminal's records of the total number and value of SNAP, credit, and total transactions **according to the terminal** during the market day, and must be copied manually. The lower table contains an automatic calculation of the same values **according to the device**. The two tables must be equal before the market day can be closed, i.e., the terminal totals must match the transaction totals on the device.

If the values from the terminal and the device do not match, the most likely reasons are a typo or a misrecorded transaction. There is no way to know which device has the error, so you must look at the transactions recorded on each.

Creating a new transaction will invalidate the terminal totals if they've already been completed.

Closing the market day

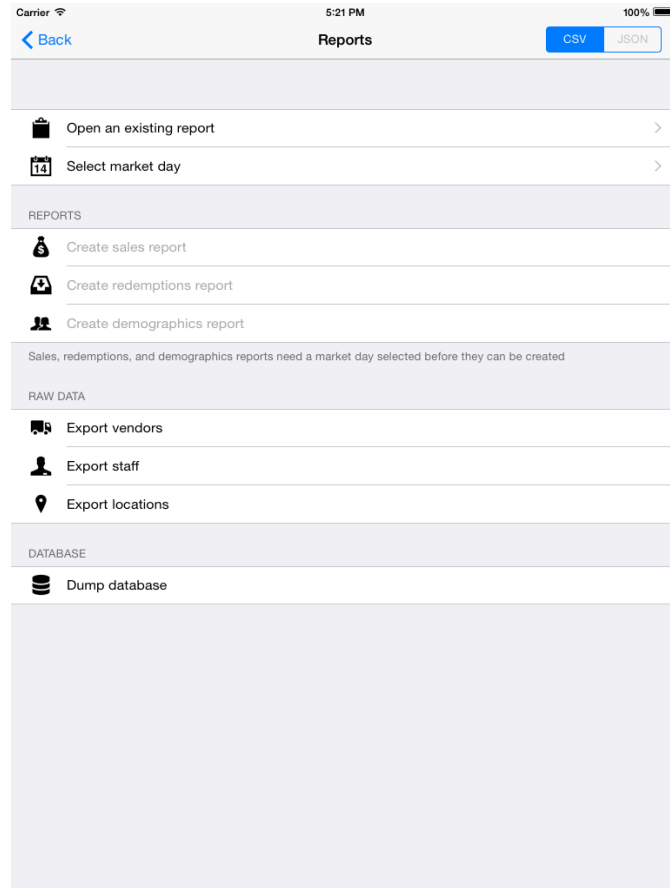
After reconciling terminal totals, you can close the market day. You may reopen the market day at any time to come back to it, to view, add, or edit redemptions or transactions, by going to the main menu and selecting *Reopen a market day*.

Managing vendors, staff, and locations

From main menu, press *Edit Vendors*, *Edit Staff*, or *Edit Locations*. You can press the + icon at the upper right to add a new item, tap on the list to edit one, or swipe across and press *Delete*

to delete one. A vendor, staff member, or location that is being used in a market day in the database cannot be removed.

Reports



Reports are used to export data in the database to a spreadsheet, where further calculations can be performed. They can also be easily viewed or edited on the device or on a computer with a spreadsheet application like Excel or Numbers.

To create a sales, redemptions, or demographic report, press *Select a market day*, select a market day, and then press the report type you would like to create.

Managing reports

To view any report on the device, press *Open an existing report*. The reports are sorted by market day, and the type of report is highlighted to make finding the correct report easier. Tap the report you'd like to view to open a preview screen. From this screen you can press the *Export* button in the top right to send the report to another program.

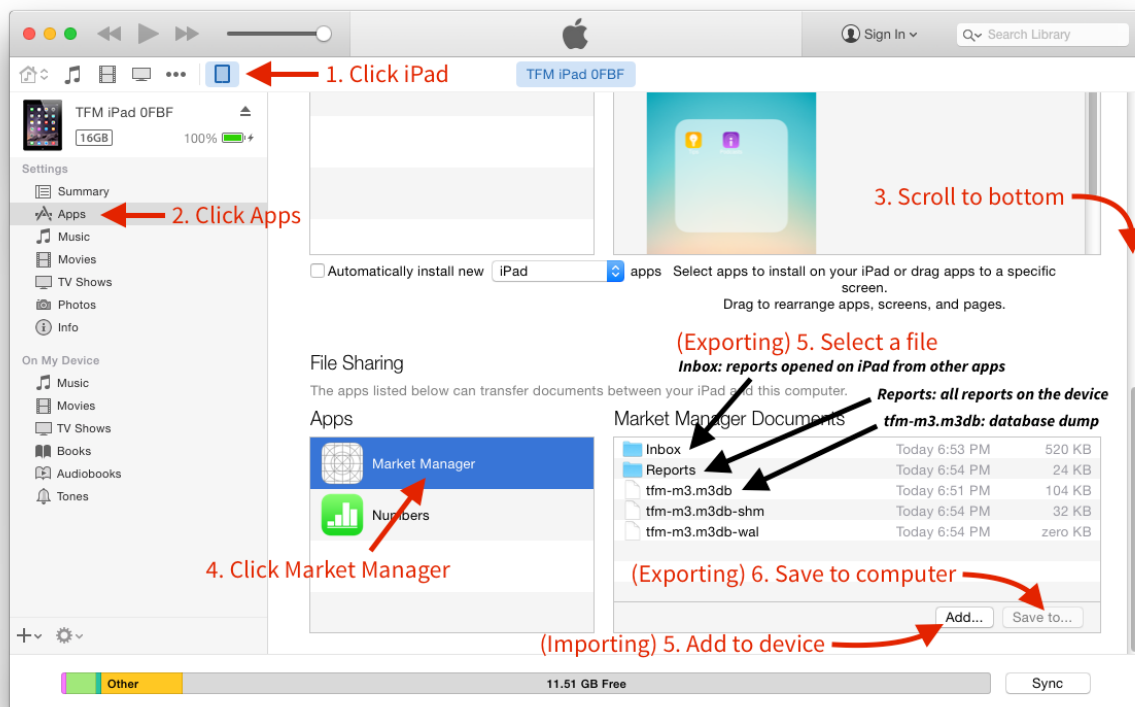
Raw data reports & database dumps

Raw data reports do not need a market day selected in order to be created. These reports do not do any kind of processing or calculations, and are generally used to simply export data from the database for backup, recordkeeping, or importing on to another device later.

Database dumps are binary data, meaning they are immutable and can't be directly edited, but they are the best way to back up the device's database in its current state. A market day does not need to be selected in order to dump the database. Market days, transactions, redemptions, vendors, staff, and locations are all included in the dump, and that data and all of its relationships are retained and are immediately transferable to another device.

Managing data through iTunes

(Make sure [iTunes](#) is installed first.)



0. Connect the device to a computer and open iTunes
1. Click the iPad icon in the top navigation bar
2. Click the *Apps* section in the left panel
3. Scroll to the bottom to the *File Sharing* section
4. Click the *Market Manager* app

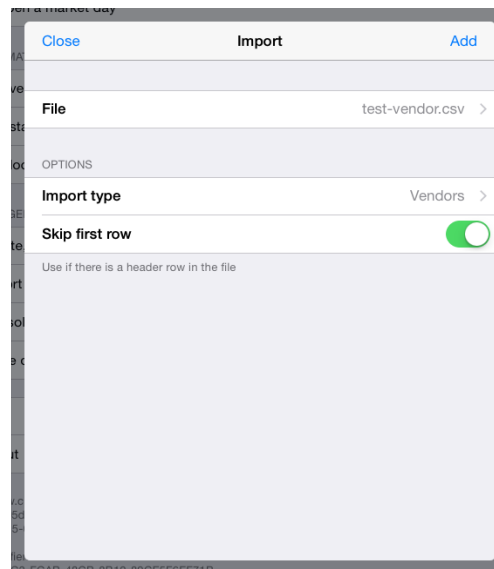
Adding data to the device

5. Press *Add* to add the data you'd like to put onto the device
 - iTunes does not let you add files to a folder, so the data must be uploaded into the main documents folder.
6. Press *Sync*
7. Open Market Manager on the device and select *Import data* on the main menu
8. Select the files you'd like to import and press *Import*
 - Only one file can be added at a time; there is no way to add multiple files at once to the database

Taking data off the device

5. Select the *Reports* folder
6. Press *Save to...* to save the data to the computer
 - iTunes does not let you open folders on the device and save individual files from them, so the whole folder of reports must be saved at once.
 - Be sure not to overwrite an existing Reports folder on your computer. You should rename the Reports folder once you've transferred it off of the device to keep from accidentally overwriting it.
 - Saving reports doesn't remove them from the device. To remove reports from the device, open the *Select an existing report* menu option in the *Reports* menu and swipe across a report you'd like to delete and press *Delete*, or press *Delete All*.

Importing data



To import data to the device, open a supported file type (`csv`, `m3db`, or `m3table`) on the device and open it in Market Manager. You can also upload a file to import into the app's document folder through iTunes. (Steps are detailed in the previous section.)

The app detects the destination automatically based on the filename, but if necessary, change the destination of the data using the picker (*Import type*). If you are importing a CSV

and there is a header row, make sure the *Skip first row* switch is turned on (**it is on by default**). If there is no header row, be sure to switch it off. Press *Import* to add the data to the database.

Importing a `csv` or `m3table` will always create new records, which might result in duplicate data. Importing a database dump (`m3db`) will add the data in the dump with the data that already exists in the database, which may also create duplicates. If you would like to avoid creating duplicates using a CSV, you must edit the data before importing it. To avoid creating duplicates with a database dump, your only options are to manually delete duplicates on the device before or after the import, or to completely reset the database before importing.

Import format

The importer is not very smart. It expects imported data in the order specified in the tables below. It will reject any data that does not have the right number of columns, but does not know if the data it is importing is actually what is in the column.

For text values, make sure the data is enclosed in quotation marks (e.g. `"text"`), so that any commas or spaces that are inside will not be interpreted as the start of the next column.

“Yes” or “No” values are not case-sensitive, and can be replaced with 1 or 0, “y” or “n”, or “true” or “false”.

Vendors

Business Name (text)	Product Types (text)	Operator Name (text)	Address (text)	Phone (text)	Email (text)	Accepts SNAP (“yes” or “no”)	Accepts Incentives (“yes” or “no”)	State Tax ID (text)	Federal Tax ID (text)
-------------------------	-------------------------	-------------------------	-------------------	-----------------	-----------------	---------------------------------	---------------------------------------	------------------------	--------------------------

Staff

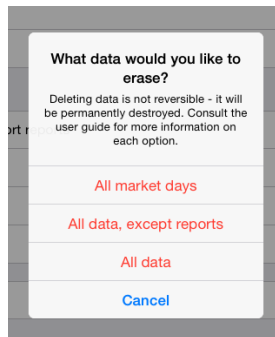
Name (text)	Phone (text)	Email (text)	Position (integer)
-------------	--------------	--------------	--------------------

See [Enumerated value reference](#) for valid values for position.

Locations

Name (text)	Address (text)
-------------	----------------

Erase data



To erase data on the device, press *Erase data* on the main menu. From this dialog, you can erase just market days, all data excluding reports, or all of Market Manager's data stored on the device. Pressing an action will delete the data immediately and permanently, so use caution.

All market days

Erasing market days will only remove market days and their associated transactions and redemptions.

All data, except reports

Erasing all data except reports will clear all market days, transactions, redemptions, vendors, staff, and locations. Reports are not stored in the database and will not be modified. Only use this after transferring all data off of the device, so that no data is lost.

All data

Erasing all data including reports will completely empty the Market Manager app of all data, resetting it completely.

Using the console

The console is intended primarily for diagnosing database errors, and for performing arbitrary queries that aren't or can't be included in reports. SQL knowledge may be needed to use this window, and referring to the [NSPredicate syntax documentation](#) is highly recommended.

Developer Documentation

Project settings

- Created with Xcode version 6.2 with 3.2-compatible format
- Tested on iPad Simulator and 2nd-generation iPads (iPad 2) with iOS 8.0, 8.1, and 8.2
- Read-only Git repository at <http://repos.ael.me/TFMMobileMarketManager>

Project file structure

- Assets.xcassets/
 - Image files used in the program
- CHCSVParser/
 - CSV parser used for data import
 - <https://github.com/davedelong/CHCSVParser/>
- FXForms/
 - Super-easy-to-use forms library
 - <https://github.com/nicklockwood/FXForms/>
- ImportTool/
 - Library used for importing data to the database from CSVs
- ReportGenerator/
 - Library for generating reports from the database as CSVs
- storyboards/
 - Storyboard layouts for UI flow and appearance of some windows
 - Other windows' appearance is handled automatically by FXForms, or their view controller file in views/
- types/
 - Data structures for entities in the database
- UITextField+PrefixSuffix/
 - UITextField category used for displaying prefix and suffix labels on the sides of text inputs
 - <http://stackoverflow.com/a/12267979>
- views/
 - View controllers and some XIBs for the user interface
 - Organized by role
- tfm-m3.xcdatamodeld
 - Database model

Enumerated value reference

(Includes type definitions created specifically for the Market Manager app, and not those used in included libraries.)

Types used in reports

Ethnicity (Transactions.cust_ethnicity)

Ethnicity	Value	Typedef Value
White	0	EthnicityWhite
Black	1	EthnicityBlack
Hispanic	2	EthnicityHispanic
Asian	3	EthnicityAsian
Other	4	EthnicityOther

Frequency (Transactions.cust_frequency)

Frequency	Value	Typedef Value
First time	0	FrequencyFirstTime
Few times a season	1	FrequencySeason
Monthly	2	FrequencyMonthly
Every few weeks	3	FrequencyNQWeekly
Weekly	4	FrequencyWeekly

Gender (Transactions.cust_gender)

Gender	Value	Typedef Value
Male	0	GenderMale
Female	1	GenderFemale
Other	2	GenderOther

Position (MarketStaff.position)

Position	Value	Typedef Value
Volunteer	0	PositionVolunteer
Manager	1	PositionManager
Accountant	2	PositionAccountant
Administrator	3	PositionAdministrator

Internal types

ImportType

Database destination	Value	Typedef Value
Database merge/replace	-1	ImportTypeDump
Vendors	0	ImportTypeVendors
Staff	1	ImportTypeStaff
Locations	2	ImportTypeLocations

ImportFormat

File format	Value	Typedef Value
Comma separated values	0	ImportFormatCSV
JSON	1	ImportFormatJSON

ImportDumpOptions

Sync method	Value	Typedef Value
Replace all data with dump contents	0	ImportDumpReplace
Merge contents of dump with current database	1	ImportDumpMerge
Update records that exist in device database from dump; do not add records that are not already on the device	2	ImportDumpUpdate

To-do list

User interface

- (UI) Grouping in various table views
 - Market days should be grouped by location (alphabetically) then sorted by date (newest first)
 - Vendors should be grouped by active/inactive (`[MarketDays count] > 0`) then alphabetically by the business name
 - Staff should be grouped by position, then alphabetically
 - Transactions should be grouped by hour in ascending order, then sorted by time in ascending order
 - Redemptions should be grouped by paid/unpaid (`[check_number > 0]`) then by business name
- (UI) Fix `textDetailLabel` in `ExistingReportsViewController` so that file modification date isn't depending on `textLabel.text` to search for filename

- Change table storage from {array within dict} to {dict within dict} containing name, filename, and file creation date (rather than just name as it is currently)
- (UI) Only show prompt to close on forms when data has been changed
 - Create new managed object context
 - Create new managed object with nil values in context, or grab from db if in edit mode
 - Perform changes directly on object, live while editing
 - Remove save button, only have Close button
 - If `[managedObjectContext hasChanges]`, display a `UIAlertController` with 3 actions: *Save and close*, *Close without saving*, and *Don't close* (cancel)
 - Else just close without a prompt
- (UI) Add `-description` method to all custom data types
 - Some already have `-description` method added, but not all
 - Mainly for better display in the console and easier report generation

Core functionality

- (Core) Don't use app delegate's managed object context as a global, create a new managed object context when needed and close it properly
 - Will also help in determining if there are changes specific to the view
- (Core) Ask to merge database when importing a database dump, instead of only overwriting
 - I think drop-in replacements is straight-up broken anyway?
 - <http://mikeabdullah.net/merging-saved-changes-betwe.html>
 - <http://stackoverflow.com/a/6959868>
- (Core) Save market open/closed state on `-applicationDidEnterBackground` and `-applicationWillEnterForeground` in AppDelegate.h
 - temporarily save `activeMarketDay.objectID` somewhere
- (Core) Fix dangerous assertion in `-handleOpenURL` that the app's first view controller will be `MainMenuViewController`'s navigation controller and has a segue named `ImportSegue`
- (Core) Export to JSON
 - `NSJSONSerialization`
 - Need to serialize classes first
 - `NSCoder`
 - `[CoreData saveToURL]`
- (Core) Import from JSON
 - Add `-importFromJSON:` methods to `ImportTool`
 - Refactor CSV import methods to `-importFromCSV:skipFirstRow:`
- (Core) Data synchronization
 - Not possible over USB due to Apple's hardware policy
 - May require integration with Dropbox or other sync service
 - try to not depend on third-party services; keep in-house
 - check in/out db from git/svn repo on central PC

■ <https://github.com/drewmccormack/ensembles>

Low-priority

- (Low priority, UI) Add preferences window and keep settings in Info.plist
 - Preferences for default values for forms
 - User-specified device name (instead of using `[[UIDevice currentDevice] name]`)
- (Low priority, UI) Address QLPreviewController segue transition warnings
 - Preview controllers throw warning when presented: Unbalanced calls to begin/end appearance transitions for QLRemotePreviewContentController
 - <http://stackoverflow.com/a/26310000>
 - <http://stackoverflow.com/q/14412890>
- (Low priority, Core) Encrypt state and federal tax IDs for vendors, since they are personally identifiable
 - NSValueTransformer
 - <http://stackoverflow.com/a/1646605>
 - <http://www.artandlogic.com/blog/2012/07/securing-your-core-data-with-transformable-attributes/>
- (Low priority, Core) Refactor type values to consistently use camelCase instead of underscores to separate words
 - Transactions, TokenTotals, and TerminalTotals use underscores and shouldn't
- (Low priority, Core) Should we be using Core Data at all if we want to be able to run arbitrary queries on another platform?
 - <http://www.objc.io/issue-4/SQLite-instead-of-core-data.html>
 - <https://github.com/ccgus/fmdb>
 - <https://github.com/marcoarment/FCModel>
 - Some initial progress was started but left aside, see `fmdb` branch in git repo