cashflow Design

# Purpose

The reasons I keep a budget are:

* To ensure that 10% is given to Charity
* To save money for big purchases or goals
* Maintain an appropriate emergency fund
* Monitor account transactions for fraud
* To implement the 10% rule

# Cashflow Ideas and Concepts

Future:

* Maybe in the future I only need a few envelopes (funds): Charity, Bills, Goals (fund for each Goal), Spending
  + Categories: Bank, Spending, Charity, Goals
  + Funds: Bills, Spending, Charity, [fund for each goal]

Future concept:

* Each paycheck, contribute a fixed dollar amount or percentage to special funds / “envelopes”. These add up to less than 100% of paycheck:
  + Charity
  + gifts
  + travel
  + home improvement
  + new car
  + car expenses
  + Emergency Fund
* Create a budget based on monthly income that accounts for bills and distributions to Funds (envelopes). Update this budget each time salary changes, and adjust how much is allotted to each Fund if needed.
* Money not allotted to funds is all pooled together and available for whatever. Bills will come out of here, as well as spending and living expenses.
* Project a desired account balance to account for fund balances (emergency, charity, and all funds and goals). Calculate balance of all accounts and compare to desired balance to determine spending power.
* Record the following transactions: Charity, Emergency, all other Funds and Goals
* Do NOT record the following transactions:
  + Bills
  + Spending and Living
* Balancing accounts
  + log in to bank accounts periodically and review transactions for accuracy. Enter balance of each account, and the date of that balance, in ledger. No need to record each transaction, just the balances, as well as any transaction that counts against a Fund.
  + When balance is entered, system calculates the amount of unrecorded transactions and enters it as a line item in the “General” category.
* Question with this: Do I want to keep recording transactions by hand in order to verify
  + I can also enter transactions against the General category by hand if I want to, in order to record the purpose of it. The line item that is inserted automatically when a balance is taken will account for any items not entered. If I later add a transaction for a time period that was already balanced, the system will automatically adjust for it next time an account balance is entered.

# System Concepts

* Make an incremental change and just stop recording a couple of the biggest categories
* For example, stop distinguishing Spending from Living, and make a separate Goal category for gifts and big spending. So I’d be putting day-to-day spending in with Living and keeping more occasional spending separate.
* Gifts is already a fund, add some % distribution to it for now

Process

* Record all transactions except the Living category.
* When I balance my accounts, calculate the difference between the account balances and the budget balances
* Enter a transaction for Living equal to the difference calculated.

Future:

* Create tools to read in exported files from Chase, Widget, and AmEx and compare those transactions to the ones entered by hand, using the account, amount, and date (+/- 3 days) information. Line up transactions from bank with transactions entered, and identify any differences.

# Requirements

## Definitions

**Funds** – These are like the envelopes. Money is spent out of Funds. Each Fund is in a Category, and money spent from that Fund is deducted from the corresponding Category. For example, the Grocery Fund is in the Living category. When groceries are bought, the money is deducted from Living.

**Categories** – These are groups of funds, or groups of envelopes. Distributions are paid to Categories, not to Funds. For example, Living is a Category, and it includes Funds Grocery, Gasoline.

**Goals –** Goals are like temporary envelopes. They may receive distributions, so in that way they are similar to Categories. Is there a need to group Goals into categories as well, or associate them with a category? Should money for a goal come from distribution, or should it be transferred out of a category? Answer: create a Category called “Goals,” and pay Distribution to that. Specific goals will be handled like Funds, and their Category will be set to “Goals. So Goals is really a Category, and it contains a special type of Fund. What makes it special? It has an Active property, and it has a Target. Could I just do that for all Funds?

## Functions

* Enter transaction data
  + Allow user to insert new transactions above anywhere in the list
  + Mobile-friendly option
  + Three types:
    - Standard transaction (income or expense)
    - Transfer between Funds
    - Transfer between Accounts
    - Transaction split between multiple funds.
      * Saved as multiple transactions with the same sort order. Anytime sort order changes, all transactions have to move together.
      * Displayed differently: one date and description, but multiple amounts and funds
* Archive transaction data
  + When transactions are archived, save the balance of each account and fund
  + Allow user to manually enter the balance of an account to compare to the calculated balance.
* Allow user to change funds and categories
  + Set goals
  + View goals
  + Goals are stored in a flexible format
  + Goals can be completed and archived.
  + Funds can have target balances
  + Funds and categories are essentially the same thing. They are both envelopes that are intended to be used for a while. The both have optional target balances.
  + Goals also have a target amount, and they have an Active property that can be set to TRUE (Active) or FALSE (inactive)
  + Inactive goals are not loaded or sent to the client in most cases. The user has the option to view “Completed Goals,” in which case only the inactive goals are sent.
  + Goals have a “Date Completed” property that is populated with the current date when the Active property is set to False. This can be shown to the user when Completed Goals are displayed.
* Set Emergency fund (FUTURE)
  + Outcome drives the target for the fund called Emergency Fund
* Allow user to set Distribution amounts and effective date.
  + Income can be distributed to Categories or active Goals.
  + User can change distribution percentages through the web interface
  + User can see current and historical distribution amounts; so it might be best not to store those as properties of the Funds or Goals.
    - Option 1: Each Fund/Category and each Goal has a property called Distribution.
      * 1a: This property is a structure with two columns: effective date, and percentage
      * 1b: This property has only the value from the most recent Distribution. Historical values are kept somewhere else.
    - Option 2: Create an array of structures called Distribution. Each structure in the array has a field for effective date, and for each Fund/Category and each active Goal. When a new goal is added, part of the process includes adding it to the most recent Distribution structure. When a goal is set to Inactive, nothing happens immediately, but that Goal is not available as an option the next time Distributions are set. This means each structure in the Distribution array may have different fields.
    - Selection: Option 2 because it is most similar to the Excel implementation and it is cleanest because it keeps all of the information in one place that is easy to reference.

### dashboard4.php

Display all Account and Fund balances

* Last three entries in cash\_balances, and balance after all recent transactions

## Data to Store

Structures and properties

* List of Funds and Goals
  + Active
  + Balance – Funds don’t have balances, Categories do. But then does that mean each Goal needs to be a category?
  + Category
    - for all goals, Category=Goals
    - many funds will be in a category of the same name. For example, Home Improvement is both a fund and a category; it is the only fund in that category.
* Distributions
  + Current and historical distribution percentages
* Account Names and Balances

# Design

## Data Structures

Non-bullet items are variable names for the structures. Bullets underneath are properties of the structure.

Funds

* Bills
  + Active = TRUE
  + Balance =
  + Category = Bank
  + Target = [ ]
* Loans
  + Active = TRUE
  + Balance =
  + Category = Bank
  + Target = [ ]
* Bank Other
  + Active = TRUE
  + Balance =
  + Category = Bank
  + Target = [ ]
* Grocery
  + Active = TRUE
  + Balance =
  + Category = Living
  + Target = [ ]
* Dining
  + Active = TRUE
  + Balance =
  + Category = Living
  + Target = [ ]
* Gasoline
  + Active = TRUE
  + Balance =
  + Category = Living
  + Target = [ ]
* Spending Other
  + Active = TRUE
  + Balance =
  + Category = Spending
  + Target = [ ]
* Charity
  + Active = TRUE
  + Balance =
  + Category = Charity
  + Target = [ ]
* Marann / TRUE / / Marann / [ ]
* Ten Percent / TRUE / / Ten Percent / [ ]
* Emergency / TRUE / / Emergency / [ ]
* Gifts / TRUE / / Gifts / [ ]
* TEAM / TRUE / / TEAM / [ ]
* Wedding / TRUE / / Wedding / ($200x5 + shower gifts)
* etc… (see Names tab of Cashflow7.xlsx)

Distribution

* [0]
  + DateEffective = 9/13/2010
  + Bank = 0.63
  + Spending = 0.27
  + Charity = 0.10
* [19]
  + Date Effective = 3/15/2017
  + Bank = 0.40
  + Spending = 0.06
  + Charity = 0.10
  + etc… see Distribution tab of Cashflow7.xlsx

Accounts (keys = name)

* Cash
  + balance
  + dateBalanced = 4/14/2017
* WidgetSavings
  + balance
  + dateBalanced = 4/14/2017
* etc (see Cashflow tab of Cashflow7.xlsx, or G2:G8 of Names tab)

### Object Classes

The above data structures are implemented as an array of objects.

|  |  |
| --- | --- |
| Object | Properties |
| fund | active  balance  category  target |
| distribution | *all elements of categories\_list* are declared here |
| account | balance  date balanced |

The variable *Funds* is an array of objects of class ‘fund.’

The variable *Distribution* is an array of objects of class ‘distribution.’

The variable *Accounts* is an array of objects of class ‘account.’

## Data Storage

### Names

Store names of all accounts, funds, categories, and goals

|  |  |  |
| --- | --- | --- |
| accounts\_categories | | |
| Name | Type | Other Names |
| Cash | Account |  |
| Widget | Account | GE Checking |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| funds\_goals | | | | |
| Name | Type | Active | Category | Goal |
| Grocery | Fund | TRUE | Living | 0 |
|  |  |  |  |  |

### Distribution

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| distribution | | | | |
| IDX | Date | Bank | Spending | … |
|  |  |  |  |  |



## Scripts and Functions

### databaseConnect.php

Short php file to connect to the database and create a mysqli object.

### Other scripts

UI with form to add a transaction

Mobile-friendly UI for adding a transaction

PHP to perform the commit to the database

* Includes taking action based on Sort Order input (input may be Before item X)

Display all recently added transactions (not archived).

* Include an “Add Before” button at the end of each row to add a transaction with that sort number
* Include new balance of accounts and/or funds affected (may be 2 accounts, 2 funds, or one of each)

Display selected account or fund (like balance3 today)

* Includes updated balance of the account or fund on each row

Display all archived transactions that meet certain search criteria (Fund, Account, or Date range)

Archive transactions

* includes updating balances in cash\_balances

### names.php

Reads accounts\_categories table and funds\_goals table. Creates the following arrays:

* accounts\_list
* categories\_list
* funds\_list

### classes.php

Declare classes *fund*, *distribution*, and *account* according to [Object Classes](#_Object_Classes).

### balance.php

Used for balancing accounts. Focuses on account balances, but may also display category, fund, and goal balances. Includes buttons for adding transactions, changing sort order, and archiving transactions.

#### Default unfiltered

Top section: display all account balances. each number is a link that will filter the page for that account.

Next: display all new (unarchived) transactions, with most recent first. include account balances after each transaction. Future: also display the balance of the category(ies) that changed

* call read\_cash\_balances\_idx() for balances from most recent archive
* call read\_new\_transactions() to read all unarchived transactions from the database and return an array of transaction objects
  + for each row in the result, call read\_transaction() to return transaction as an object
* for each transaction object in the array, call process\_transaction() to adjust balances of Accounts and Categories. Display the result in a table on the page.

Next: display form for adding a transaction

Next: display buttons:

* Update
* Archive
* Delete
* Move 2nd above 1st
* Move multiple
* Select All

Next: display filters for archived transactions. default is the previous 30 days.

Next: display archived transactions in a table, including only those that meet the filter criteria

#### Account Filter selected

If one of the account balances is clicked, the page is reloaded with a variable sent by URL. Most of the page layout is the same, but it is focused on one account.

Top Section: Display balance of this account, centered. Immediately below it, display a link to View All; if clicked, reload page according to [Default unfiltered](#_Default_unfiltered).

Next: display all new (unarchived) transactions that affect the selected account, with most recent first. Include balance of selected account after each transaction. Future: also display the balance of the category(ies) that changed.

Next: display form for adding a transaction, and use the selected account as the default setting for the drop-down.

### read\_transaction()

Input:

Output: transaction object

### build\_distributions()

creates an array of objects of class “distribution” with fields for each Category that gets money. Each object in the array is a different entry in the “distributions” table and has a unique date. Each field in each object is a category.

Syntax example: $distributions[0]->date, $distributions[0]->Bank, etc.

### process\_transactions()

Input: transaction object or array of transaction objects, account balances (Accounts), category balances (Categories)

Output: updated account and category balances (Accounts, Categories)

# Next Steps

1. Create balance.php
   1. debug process\_transactions and print\_ledger\_row
   2. Consider layout:
      1. add button to hide the ledger, or hide it by default and add a button to show it. Pass value of button to process\_transactions.
      2. Or I could pull print\_ledger\_row up one level to balance.php and have process\_transactions only process one transaction. In that scenario I would call process\_transactions from within a for loop in balance.php. Then after it had all been calculated, I would display the balances. Then after the balances, I would re-run process\_transactions in a for loop, and this time I would display the ledger as I go by calling print\_ledger\_row in the same forloop in balance.php.
      3. probably just hide the ledger on balance.php. It is more for debugging right now. Show it on a different page.
2. Design and document workflow for archiving

Here’s the bottom of balance.php as of rev 24 of functions:

idx = 0 and cat is Spending Other  
idx = 1 and cat is Marann  
idx = 2 and cat is Card  
idx = 3 and cat is Bills  
idx = 4 and cat is Gifts  
idx = 5 and cat is Living  
idx = 6 and cat is Spending Other  
idx = 7 and cat is Gasoline  
Sum of Distributions does not equal 1. It is 0.94  
  
Using idx 0 for Income distribution on transaction 8  
  
2017-03-15 =   
0.4 =   
0.06 =   
0.1 =   
**Fatal error**: Cannot access empty property in **/srv/disk7/1544017/www/bryanhermsen.com/cashflow2/cash2\_functions.php** on line **412**

## To Do List

1. (done) Create funds\_goals table in database (done), add all funds and goals currently in use
2. (done) Should I change array keys to the names for Accounts, Funds? Instead of Accounts[1], Accounts[Cash], or Accounts[Widget\_Checking].
   1. Yes, do that.
3. (done) What is the plan for Categories? There is no Category class right now. Review spec above and decide if I need to create one in names.php and pass it to read\_cash\_balances\_idx(). Otherwise I’m only updating the funds, which is stuff like Grocery, Bills, etc.
   1. I do need a Categories class.
   2. Added it to names.php and read\_cash\_balances\_idx().
4. Create function for processing transactions that can be called from dashboard4.php and from the function that adds things to the database.
   1. cash2\_functions/read\_transaction
      1. NEXT: create and return transaction object
   2. cash2\_functions/process\_transaction
5. Create function for archiving transactions
6. Complete dashboard4.php
   1. Rename all documentation from .html to .php
   2. (done) Continue moving the “/\*” down the page and troubleshooting as I go.
   3. (done) Test and troubleshoot include for classes.php (done)
   4. test and troubleshoot names.php
   5. Read last three entries from cash\_balances
   6. Display last three entries from cash\_balances
7. Finish classes.php (distribution class)
8. Create balance.php

# Appendix

From Todoist:

1) Adjust distributions  
- Emergency 2%  
- Adjust bank based on bills  
- Consider combining Spending and Living; maybe reopen Gifts as a separate category? Has there been any benefit to keeping Living expenses separate from Spending Money? I guess the benefit is that I could set aside my own spending money apart from Groceries and Dining. Consider rearranging and reallocating.  
- Create goal for car  
  
2) Create Goals as temporary categories   
3) Decide what to stop recording. Maybe Spending and Living? Maybe stop recording Bank and assume it will follow the budget each month?  
4) Specify way to do balancing/accounting. Use account balances to make assumptions about spending in unrecorded categories.