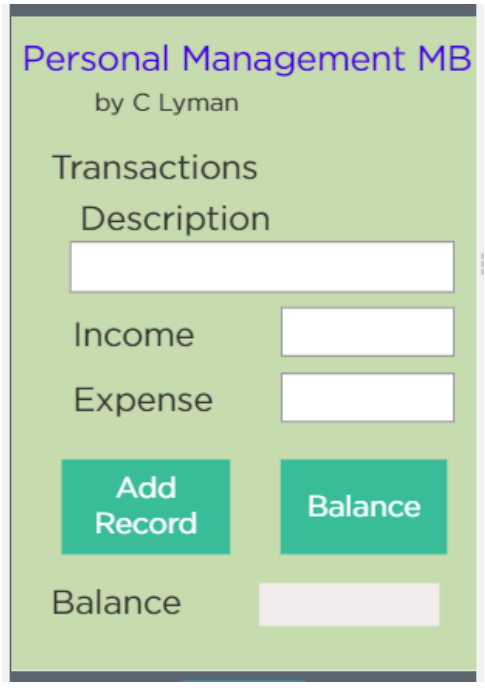
Personal Management MB app

The ability to keep a record of the income and expenses for 3 months for the Personal Management MB can be much easier by creating an app that uses an online database. The app can be created in Code.org's App Lab and installed on a phone or tablet then whenever a money is spent or received the transaction can be enter in the app. This process can eliminate one of big problems with getting the merit badge, keeping the money record for 3 months.

This app can be done in 4 parts. 1) Designing the screen for the app, 2) Coding the entering and storing the transaction in the online database, 3) Coding the retrieving the online databases records to check the current balance, and 4) Viewing and downloading the data from the online database.

Section 1: Designing the screen

In Code.org's App Lab and in Design mode the screen with its objects and properties can be designed. Each object is dragged from the objects onto the design screen. Object properties are then changed as needed. The screen background color can be changed to a light color. Placement on the screen can be done by dragging the object. Text can be changed for label objects. Font sizes are adjusted for the text input and label objects. The width of text input objects may need to be adjusted. The ID for any object that are going to interact with code need to be named appropriately.

Screen Design Plan	Finished Screen Design
	

Section 2: Coding the entering of a transaction

The code should be comment at the top and any place in the code to help understand what the code is doing. Variables are usually declared at the top of the code and initialized to the data type they will store. Examples: numerical data: `var income =0,` string data: `var description = “”.`

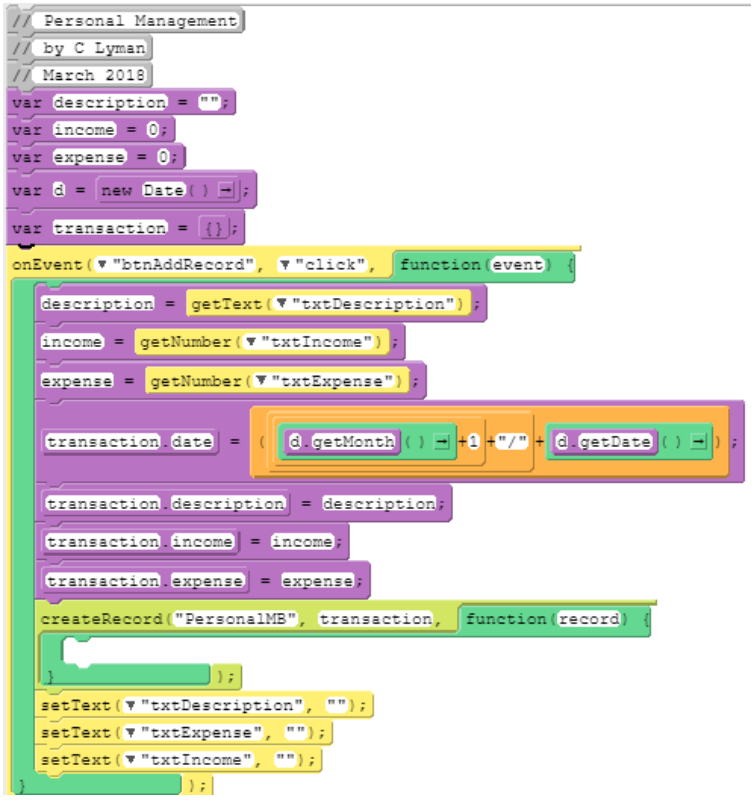
Variable objects

A special variable type can be declared as an object by using the declaration: `var transaction = { }.` This variable can hold different kinds of data, both numeric and string. This could be thought of as all the field that might be in one record of a database. Each part of the data object can be identified by the object name, a dot, and the field name. Examples of 3 data types (date, numeric, string): `transaction.date, transaction.income, transaction.description`. These are all fields or part of the variable `transaction`.

createRecord

The `createRecord(“`*`tableName`*`”, `*`recordObjectName`*`, function(record)` line of code takes 3 parameter inputs, 1) a string for the name of table 2) the name of the variable object that contains the data for the record being saved, 3) the `function(record)`. This writes the current stored data in the variable object to the online data cloud.

Block Code



JavaScript

```
// Personal Management
// by C Lyman
// March 2018
var description = "";
var income = 0;
var expense = 0;
var d = new Date();
var transaction = {};
onEvent("btnAddRecord", "click", function(event) {
  description = getText("txtDescription");
  income = getNumber("txtIncome");
  expense = getNumber("txtExpense");
  transaction.date =
(d.getMonth()+1+"/"+d.getDate());
  transaction.description = description;
  transaction.income = income;
  transaction.expense = expense;
  createRecord("PersonalMB", transaction,
function(record) {

  });
  setText("txtDescription", "");
  setText("txtExpense", "");
  setText("txtIncome", "");
});
```

Section 3: Coding the reading of the online database

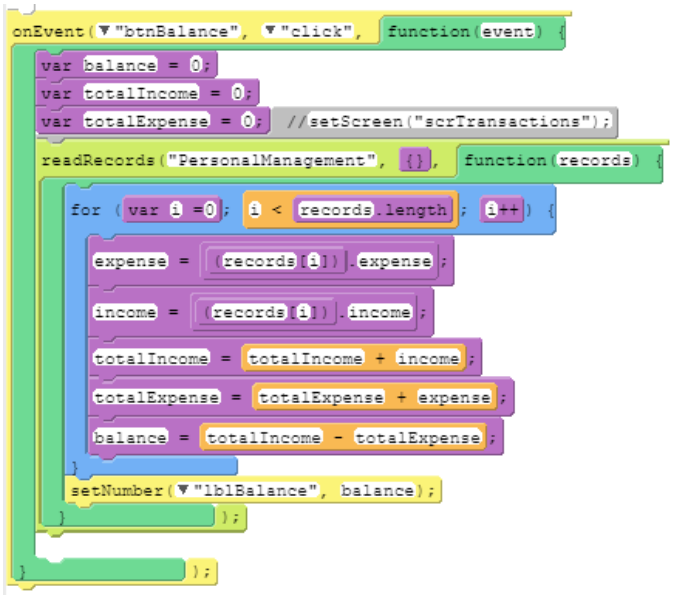
The data stored in the online data cloud database can be retrieved for use in the app. This is done with the `readRecords("tableName", { }, function(records){ …` line of code. The **readRecords** method takes 3 inputs 1) The name of the database as a string,(`"PersonalManagement"`), 2) the { } which indicate a variable object, 3) the function(records) statement. All the read data is stored in the "records" variable object. Each record can be accessed with through the `records[index]` statement. A field in the record can be accessed with the `.field` extension as in `records[i].expense` example. In the Personal Management MB the data is retrieved and ran through a for loop until last record (`records.length`) is read. As the loop is processed, the value for the income and expenses in each record are totaled. Example:

```
totalIncome = totalIncome + income;
totalExpense = totalExpense + expense;
```

In the end the total expenses are subtracted from the total income and stored in balance. Example:

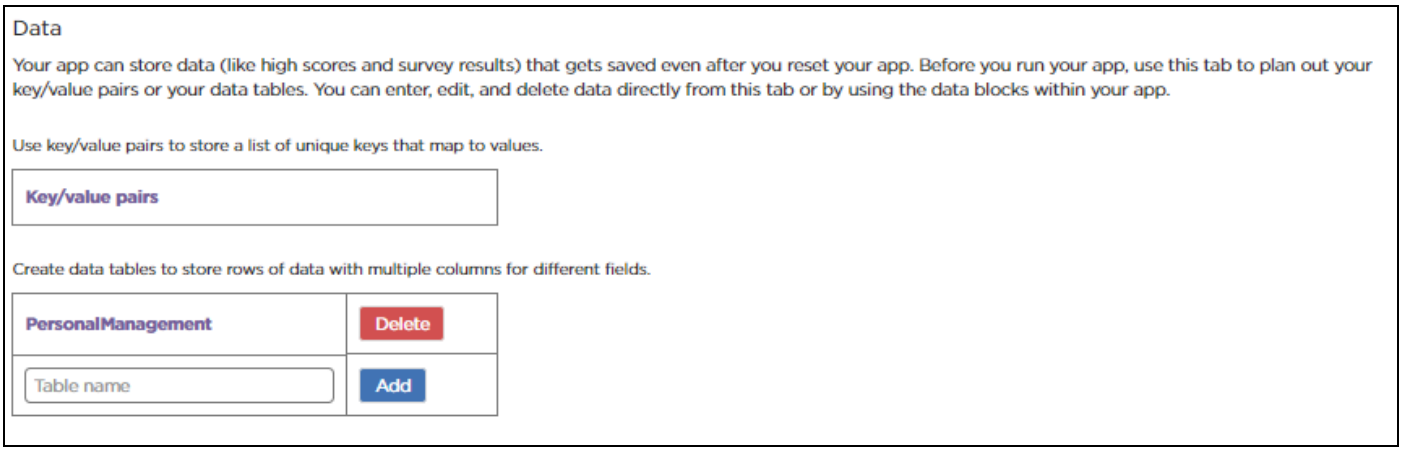
```
balance = totalIncome - totalExpense;
```

The balance is then displayed back to the screen.



Section 4: Viewing and downloading the online database

In Code.org's App Lab one of the 3 modes is the Data mode. In this mode the online database can be viewed, modified, or downloaded. Here is a screenshot of the Data dialog box. The table's name shows up in purple, "PersonalManagement".



Clicking on the database name opens the fields and records in the database. The items in the database can be viewed, edited, or deleted. In the top right is a button to export the data to a comma separated values (csv) file. This is a generic spreadsheet file. It can be opened in a spreadsheet for additional analysis if needed.