

## **Movie Inventory Maintenance Program**

### **Executive Summary**

#### **The Business**

The business is maintaining home inventory. I have older children whom I have placed responsibility on to maintain different 'zones'. One zone is the media center, where one of the children is in charge of making sure movies and video games are maintained. Part of this system requires being able to 'check-out' items to other family members. We had no system in place to easily facilitate this, so I used it as the basis of my project.

#### **System Details**

The system will maintain a list of all movies that we own. The toolbar was customized with a Movie Checkout button in the corner of the Data ribbon. When a user checks a movie out, it will track the date it is checked out, the user who checked it out, and temporarily remove it from inventory. Checking the movie back in replaces it to inventory and clears the name and date it was checked out.

The system will also be able to add new movies to the dataset. The user will enter a movie title in which the system will look up on-line and download genre information, a list of the primary actors and actresses, run time, and release date. It will then, when prompted, copy this information into the dataset.

### **Implementation Documentation**

The basic project is a table maintaining a list of movies. The table includes the following fields:

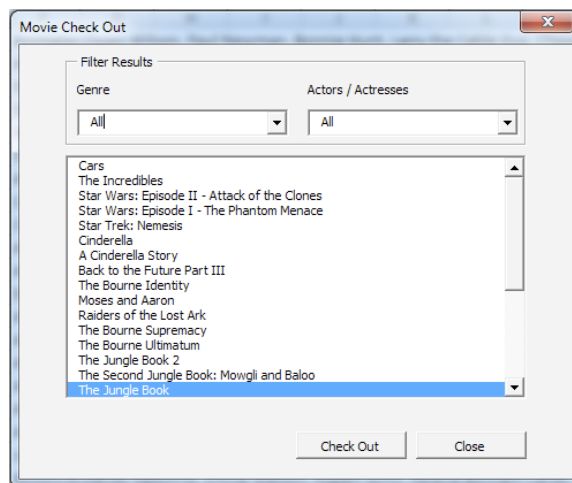
- Movie Title
- Genre – a single cell containing a comma separated list of genres assigned to the title
- Status – There are two status options; checked in or checked out
- Date – the date the title was checked out
- Name – identity of person who checked the movie out
- Run-time – the length of the movie
- Cast – a single cell containing a comma separated list of primary actors and actresses

In addition to this table, three other sheets were created for working with the data.

1. A sheet for working with movie titles. Movie titles and corresponding link information returned from a web search are stored on this page.
2. A movie detail sheet. When a specific title is selected, movie information specific to that title is downloaded to this sheet. This information is parsed through and selected data from this sheet is copied to the primary dataset.
3. A sheet for eliminating duplicate entries and sorting. This sheet is used for parsing through a list of actors and genres, eliminating duplicates, and sorting them alphabetically. This data is used for filtering records on the movie checkout form.

To facilitate working with the data on these sheets, seven forms were created.

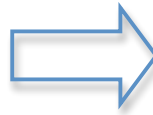
1. frmMain – the main form that loads to access the other forms.
2. frmMovie – form to facilitate checking out movies. Contains two combo boxes for filtering results, one listbox for displaying titles, and a *close* and a *checkout* button.



When this form is opened, a procedure is called that copies all of the data in the *genres* and *actors* fields from the main dataset to the *sortdata* sheet. It parses through each variable extracting individual items into a list, eliminating duplicates, and sorting alphabetically (see below for genre example).

## Genre

Animated, Comedy, Family  
Action/Adventure, Animated, Comedy, Kids/Family  
Sci-Fi Action, Science Fiction, Space Adventure  
Science Fiction, Space Adventure  
Action/Adventure  
Science Fiction, Space Adventure, Adventure  
Animated, Family, Sci-Fi/Fantasy  
Comedy, Romance  
Comedy, Drama Action/Adventure, Sci-Fi/Fantasy  
Action/Adventure, Suspense/Thriller  
Action/Adventure, Animated, Family  
Drama, Music/Performing Arts  
Action/Adventure, Sci-Fi/Fantasy  
Action/Adventure  
Action/Adventure, Suspense/Thriller  
Action/Adventure, Animated, Family

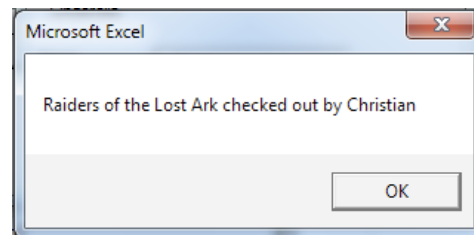


All  
Action/Adventure  
Adventure  
Animated  
Animated Musical  
Children's Fantasy  
Children's/Family  
Comedy  
Coming-of-Age  
Drama  
Drama Action/Adventure  
Family  
Family-Oriented Adventure  
Jungle Film  
Kids/Family  
Music/Performing Arts  
Musical  
Romance  
Science Fiction  
Sci-Fi Action  
Sci-Fi/Fantasy

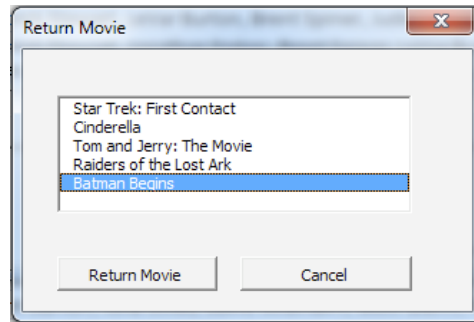
The word “All” is hardcoded in the first line of the sorted lists. These lists are then loaded into a combo box for filtering the movies by Genre or Actor.

When a movie is selected, the user clicks the ‘Checkout’ button. This calls a procedure that opens *frmCheckout*.

3. *frmCheckout* – this form has two fields, *Name* and *Date*. The name is for entering the name of the person checking out the movie, the date is the date the movie is checked out, and is auto-populated with the current date. When the button *checkout* is clicked, a procedure updates the main dataset, changing the status field from *in* to *out* and writing the name and date into the record. A *msgBox* pops up confirming the transaction.



4. *frmReturnMovie* – this form is used for returning movies. The user selects the movie (see below) and clicks return movie. A confirmation message pops up and the name and date are deleted from the corresponding record on the main dataset and the status is changed from *out* to *in*.

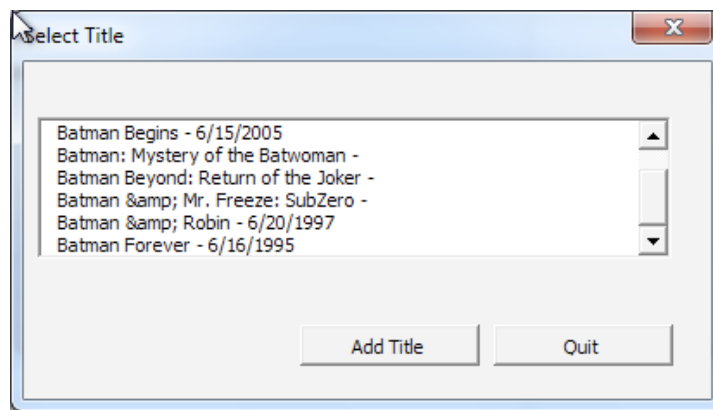


5. frmAddTitle – a text field for entering a movie title is at the top of the box. The user enters a title and clicks *Look up Movie Data*. This calls a procedure that creates a web data class that connects to movies.com to search for any matching titles and to extract the link data. It downloads the titles, link data, and release date(when available) to the *addTitle* sheet (see below).

batman/m60492	Batman	23-Jun-89
batman/m66153	Batman	
batmanbegins/m59504	Batman Begins	15-Jun-05
batman:mysteryofthebatwoman/m29414	Batman: Mystery of the Batwoman	
batmanbeyond:returnofthejoker/m24862	Batman Beyond: Return of the Joker	
batman26mr.freeze:subzero/m26974	Batman & Mr. Freeze: SubZero	
batman26robin/m59503	Batman & Robin	20-Jun-97
batmanforever/m59505	Batman Forever	16-Jun-95

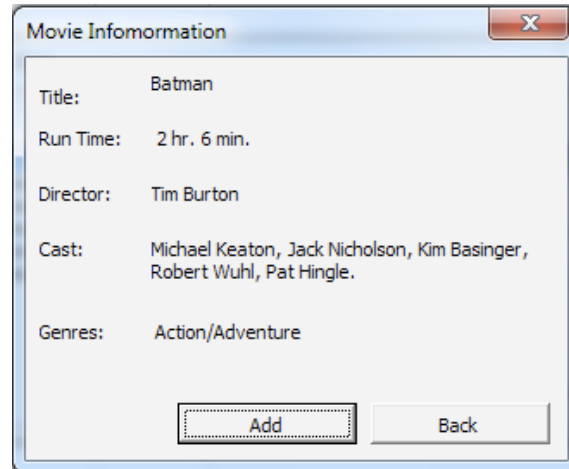
The procedure then calls *frmSelectTitles*.

6. frmSelectTitle – the data from the *addTitle* sheet is loaded into this form.



The user selects the title they want to add and clicks *Add Title*. This calls a procedure that pulls the link string from the *addTitle* sheet (see table above) and uses it to open a movies.com page (hidden in the background) and download specific title information to the *MovieDetail* sheet. Once downloaded, it calls the *frmMovieDetail* form.

7. frmMovieDetail – this form does not have any input boxes. It only contains labels to display the relevant data from the *MovieDetail* sheet for verification before adding the title to the spreadsheet.



The screenshot shows a Windows-style dialog box titled "Movie Information" with a close button (X) in the top right corner. The dialog contains the following information:

Title:	Batman
Run Time:	2 hr. 6 min.
Director:	Tim Burton
Cast:	Michael Keaton, Jack Nicholson, Kim Basinger, Robert Wuhl, Pat Hingle.
Genres:	Action/Adventure

At the bottom of the dialog, there are two buttons: "Add" (which is highlighted with a dashed border) and "Back".

When the user clicks *Add* the information is extracted from the labels and copied into the main dataset.

## Discussion of Learning

I took this class because I wanted to learn how to use VBA. I worked for several years as a database and web application developer. I was interested, however, in how to use VBA to increase the power of Excel and to automate procedures. I really enjoyed the class and learned a lot about VBA. Particularly interesting and useful is the ability to get data from the web.

Because of my programming background, however, I had a difficult time deciding on a final project. Most problems I felt could be solved better using other technologies, and I did not want to invest a lot of time in a second-rate solution. I decided to do the Movie application because it gave me an opportunity to better understand how to use Excel to communicate with the web, particularly how to extract link information from a web page. It also provided good practice using other tools I thought would be useful in the future such as organizing and filtering data, copying data across sheets, modifying macros, and customizing the toolbar. Finally, it solved an actual family need.

This project took me considerable more time than anticipated.