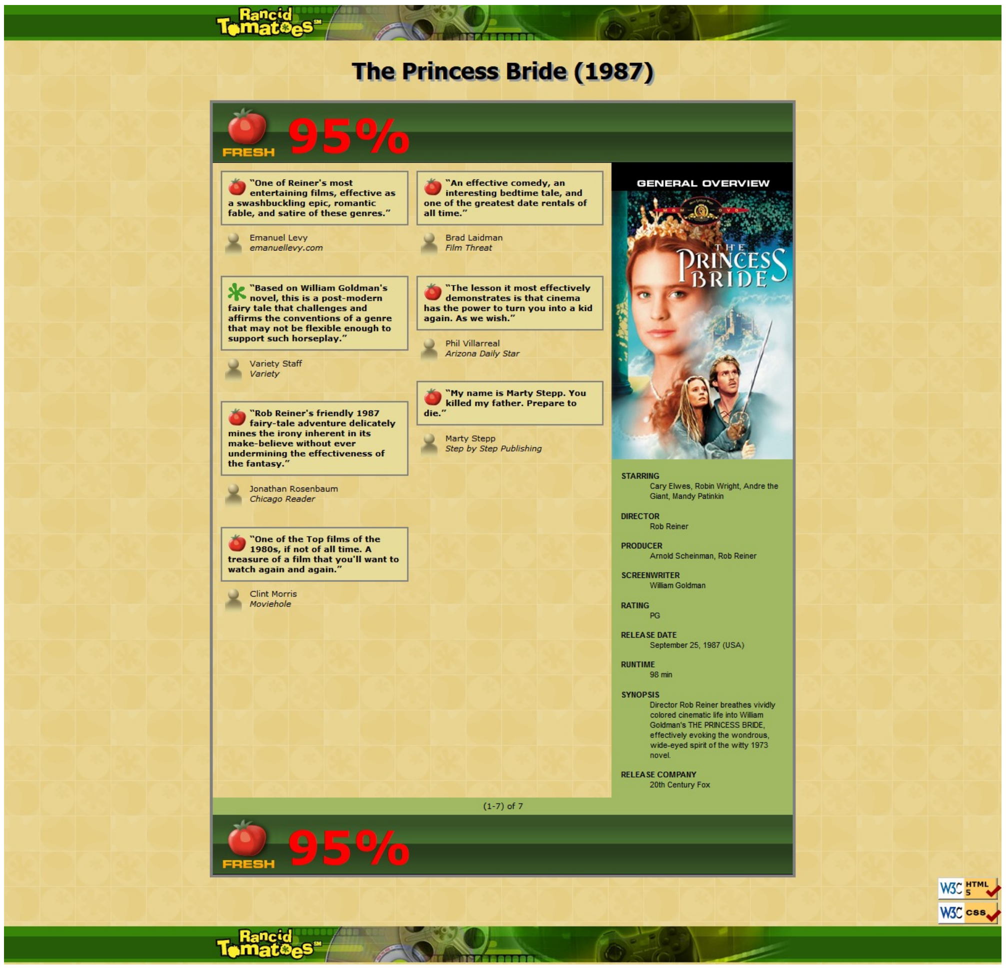
University of Washington, CSE 154
Homework Assignment 3: Movie Review Part Deux

For this assignment you will modify your HW2 movie review page using PHP code to make the page more flexible and powerful. Your PHP code will allow you to show reviews for a variety of movies using the same code. Turn in these files:

- **movie.php**, the PHP code to produce review pages for movies
- **movie.css**, the style sheet for **movie.php**

The overall page looks nearly the same as it did in the previous assignment. The main difference is that the page can show reviews and information for a film other than TMNT. For example, the following output shows The Princess Bride:



The Princess Bride (1987)

95%

"One of Reiner's most entertaining films, effective as a swashbuckling epic, romantic fable, and satire of these genres."

Emanuel Levy
emanuellevy.com

"Based on William Goldman's novel, this is a post-modern fairy tale that challenges and affirms the conventions of a genre that may not be flexible enough to support such horseplay."

Variety Staff
Variety

"Rob Reiner's friendly 1987 fairy-tale adventure delicately mines the irony inherent in its make-believe without ever undermining the effectiveness of the fantasy."

Jonathan Rosenbaum
Chicago Reader

"One of the Top films of the 1980s, if not of all time. A treasure of a film that you'll want to watch again and again."

Clint Morris
Moviehole

"An effective comedy, an interesting bedtime tale, and one of the greatest date rentals of all time."

Brad Laidman
Film Threat

"The lesson it most effectively demonstrates is that cinema has the power to turn you into a kid again. As we wish."

Phil Villarreal
Arizona Daily Star

"My name is Marty Stepp. You killed my father. Prepare to die."

Marty Stepp
Step by Step Publishing

GENERAL OVERVIEW

STARRING
Cary Elwes, Robin Wright, Andre the Giant, Mandy Patinkin

DIRECTOR
Rob Reiner

PRODUCER
Arnold Scheinman, Rob Reiner

SCREENWRITER
William Goldman

RATING
PG

RELEASE DATE
September 25, 1987 (USA)

RUNTIME
98 min

SYNOPSIS
Director Rob Reiner breathes vividly colored cinematic life into William Goldman's THE PRINCESS BRIDE, effectively evoking the wondrous, wide-eyed spirit of the witty 1973 novel.

RELEASE COMPANY
20th Century Fox

(1-7) of 7

95%

To make things simpler, the page's title should now always be just **"Rancid Tomatoes"**.

From the course web site you will download a **.ZIP archive** of input files for many movies, described on the next page. Unzip this file into the same directory as your HW3 files, so that the files will be located in relative paths such as **tmnt2/info.txt** and **princessbride/overview.png**. Your code should assume these are the relative paths to use. *(Don't put them in a* ***moviefiles/*** *folder. Put each film's folder, such as* ***tmnt2/*** *or* ***mortalkombat/****, directly in your* ***hw3*** *folder.)*

The page uses the same **images** as HW2, with two minor changes. The **tmnt.html** page showed **rottenlarge.png** next to its 33% rating. But now some films have high overall ratings. Any film whose rating is 60% or above should show **freshlarge.png**, shown at right. Also **overview.png** has moved, since each movie has its own overview image. Link to all of the "fixed" images using their full **absolute URLs**, not relative ones. The overview images are the one exception; since these now come from the support ZIP file and reside in the folder with the movie's information, link to these using a relative path such as `"tmnt2/overview.png"`.

All style elements not described here should be as specified in HW2 or subject to the preference of the web browser. The screenshots in this document were taken on Windows in the latest version of Firefox, which may differ from your system.

Appearance Details:

Your **movie.php** page will show reviews of different movies based upon a *query parameter* named `film` that is passed from the browser to the page in its URL. The value of this variable should be a string representing the movie to display. The browser will request your page with a URL such as the following:

https://webster.cs.washington.edu/*your_uwnetid*/hw3/movie.php?film=princessbride

Your PHP code can store this parameter's into a variable using code such as the following:

```
$movie = $_GET["film"];
```

All of your PHP code should use this parameter's value, and you should never hard-code particular movie names. Your code may assume that the browser has properly supplied this parameter and has given it a valid value. You may assume that the movie exists and has a corresponding folder of valid input data and images. You do not have to handle the case of a missing or empty `film` parameter, a value that contains bad characters, a value of a movie that is not found, etc.

Based upon the movie name variable's value, display a review of that film. Each film is stored in a directory named the same as your query parameter. For example, the film `princessbride` stores its files in a folder named **princessbride/** . The files associated with each movie are the following. You may assume that the files exist and are valid at all times.

- **info.txt**, a file with three lines of information about the film: its **title**, **year**, and **overall rating** percentage. The information is used as the page heading and overall red movie rating number. Here's an example file:

  ```
  The Princess Bride
  1987
  95
  ```

- **overview.txt**, a file with information to be placed in the General Overview section of your page. Each pair of neighboring lines contains the title and value for one item of information, to be displayed as a definition list term (`dt`) and its description (`dd`). The number of lines in the file varies from movie to movie. Example:

  ```
  STARRING:Cary Elwes, Robin Wright, Andre the Giant, Mandy Patinkin
  DIRECTOR:Rob Reiner
  PRODUCER:Arnold Scheinman, Rob Reiner
  SCREENWRITER:William Goldman
  ...
  ```

- **overview.png**, an image to display at the top of the General Overview section. This is the movie poster for the film, such as the one shown at right.

- **review1.txt**, **review2.txt**, ..., files containing information for each review of the film. Each review file contains exactly four lines: The review, a fresh/rotten rating, the reviewer's name, and the publication. For example, `review1.txt` might store the following:

  ```
  One of Reiner's most entertaining films, effective as a ...
  FRESH
  Emanuel Levy
  emanuellevy.com
  ```

  - Different movies have different **numbers of reviews**. Show the first half of the reviews in the left column, and the rest on the right. If a movie has an odd number of reviews, the left column receives the extra review. For example, Princess Bride has 7 reviews, with the first four go on the left and the last three on the right. If a movie has only a single review, it goes in the left column. Do not worry about the possibility that the overview and reviews sections will be wildly different in height. *(You may assume that the film has at least 1 review, but it might have 10 or more; you should display them all.)* The green bar along the bottom of the page also should now display "(1 - *N*) of *N*" where *N* is the number of reviews for the current movie.

    If a review has 10 or more reviews (such as `tmnt2`), the names will be e.g. **review01.txt**, ... So don't hard-code file names like `"review1.txt"`; look for all files that begin with `"review"` and end with `".txt"`.

  - The **image to show for each review** is affected by the second line of the review's text file. If the critic gave the film a FRESH rating, display **fresh.gif**. If it's ROTTEN, display **rotten.gif**.

For example, if your variable stores `"princessbride"`, open **princessbride/info.txt** to read the film's title/year/etc. and display that on the page. Open **princessbride/overview.txt** and display its contents in the General Overview section on the right side. Then look for all review files in `princessbride/` and display each in the reviews area.

The page also has **two of its banner sections repeated**. The top "Rancid Tomatoes" banner now appears at the very bottom of the page in addition to the top of the page. The green central bar listing the film's overall rating (such as 95% for Princess Bride) is modified to occupy the full width of the central area of the page and is repeated at the bottom of that central area. You should modify your HTML/CSS/PHP code to produce these changes in the page appearance.

We do not expect you to produce a pixel-perfect page that exactly matches our images. But your page should follow the styles specified in this document and match the look, layout, and behavior shown here as closely as possible.

Creative Aspect: Your Own Movie

As part of your turnin, create your own set of movie input data to be fed into your **movie.php** page. Write an **info.txt**, **overview.txt**, and at least 4 review text files for your movie. Also find a suitable image to use as **overview.png** of size 250x412px. You can resize an image using the free GIMP editor or a program of your choice. Turn in your movie's files as **mymovie.zip** along with your assignment.

Extra Features for CSE Majors:

In addition to the previous required features, CSE majors (B section) must also complete **all of the following** additional requirements in the movie page. If non-majors want to complete some of the extra features below, that is fine, but it is not required and will be ignored in grading. It is assumed that you completed the CSE majors' extra features for HW2 (fixed top banner, fixed W3C images, favicon, small white text after 33%, etc.) and retain those in your HW3 page.

1. **Query parameter for count of reviews:** Make your page accept a second optional integer query parameter named `reviews` that indicates the number of reviews to show. For example, if the browser requests **movie.php?film=tmnt&reviews=5**, you will show only the first 5 reviews available for that film. You may assume that if the parameter is passed, the value will be an integer. If the value is negative, treat it as though it were 0 and show no reviews. If the value is greater than the number of available reviews for that film, just show every available review for that film. For example, since there are 8 reviews for the film TMNT, **reviews=8** would have the same effect as **reviews=999**. If the parameter is not passed at all, just show all reviews for the film like usual.

   The presence of the `reviews` parameter should also change the bottom text listing the count of reviews, such as "(1-8) of 8". For example, if **reviews=5**, it should read "(1-5) of 8".

2. **Top/bottom banners fixed:** In HW2, CSE majors had to set the page's top "Rancid Tomatoes" banner to lock into a fixed position. In this assignment, retain that fixed top banner, but also add a second copy of that banner that is fixed at the bottom of the page. Both banners should remain visible on the screen and not move or scroll when the user scrolls the page. To keep the bottom banner from covering up other page content, give the page's body a bottom margin of 60px, and give the bottom-right W3C images a bottom margin of 50px.

3. **Changing page title:** Instead of having the title always be "Rancid Tomatoes", make it change based on the film. For example, if the film is "Mortal Kombat", the title should be "Rancid Tomatoes - Mortal Kombat".

4. **Error message when movie is not passed:** When the `film` parameter is not passed, or when the parameter refers to an invalid movie (one that does not have a corresponding folder in your program's directory), output an error message. The exact format of the error message is unimportant; for example, the page can just be a plain white page with a message such as, "Film not found: foobar." (But the error message page should still pass the W3C HTML validator.) If the movie's folder exists, you may still assume that the folder contains all of the necessary files to display the film and that these files contain the proper information.

5. **Meta tags in HTML:** Insert `meta` tags in the `head` section of the HTML (PHP) file to describe the page. Insert three `meta` tags: one to specify that the page's character set is UTF-8, one that gives a description of the page, and one to list some relevant keywords to search for the page. Follow the syntax from the lecture slides or Chapter 2.

   Other: Do you have an extra feature you'd like to add to your page that isn't listed here? Ask us on the course message board and we'll let you know if it is okay to substitute for one of the above.

Expected output images are provided on the course web site for the standard version of the page as well as for the CSE majors' version of the page. All of the above features can be implemented using HTML/CSS features shown in the textbook Chapters 2-6 or the lecture slides.

Development Strategy and Hints:

PHP code is difficult to **debug** if you have errors. Write the program **incrementally**, adding small pieces of code to a working page, and not advancing until you have tested the new code. We suggest that get your page to display a single film (such as `princessbride` or `tmnt2`) first, then generalize it for other films. None of your code should refer to specific names of films such as `princessbride`. The following functions may be helpful:

- `count` - returns the number of elements in an array
- `explode` - breaks apart a string into an array of smaller strings based on a delimiter
- `file` - reads the lines of a file and returns them as an array
- `glob` - given a file path or wildcard such as `"foo/bar/*.jpg"`, returns an array of matching file names
- `list` - unpacks an array into a set of variables; useful for dealing with fixed-size arrays of data
- `trim` - removes whitespace at the start/end of a string (gets rid of stray spaces in output)

Go to **php.net/*functionName*** in your browser for more information about any function.

Implementation and Grading:

Still turn in your **movie.css** even if it did not change. Our **movie.php** solution is about 120 lines long (90 "substantive"). For full credit, your HTML, CSS, and PHP code should follow the rules listed in the **Style Guide** on the class web site.

For full credit, your page's output for all films must successfully pass the W3C **HTML validator**. (Not the PHP source code itself, but the HTML output it generates, must validate. To validate your page, view the page on the server, then choose View Source in your browser and copy/paste it into the validator.) Do not use HTML tables on this assignment.

Your PHP code should not cause errors or warnings. We will also grade the **style of your PHP code**; like a CSE 14x assignment. Do not use the `global` keyword. Use indentation/spacing, and avoid long lines over 100 characters. Use material from the first three weeks of class and the first five book chapters. (Textbook Chapter 5 is a good resource.)

Make sure to test your code on all available movies from the ZIP file. You may want to think about other edge cases, such as what your page should do if the film has only a single review (it should show in the left-hand column), etc.

Be mindful of "mode switches" between PHP code and HTML tags using `<?php` and `?>` . Produce as much of your output in HTML mode as possible; only switch to "PHP mode" to compute or print dynamic content. For full credit, you should reduce the number of large complex chunks of PHP code that are placed in the middle of your HTML code. When possible, replace such chunks with functions that are called in the HTML and declared at the bottom of your file. For full credit your code must have **at least one function** to remove redundancy that would otherwise exist in the HTML. HTML or PHP code that is redundant should be placed into a function to remove the redundancy as much as possible. Recall that if code is nearly but not exactly the same, you can use functions with parameters to remove the redundancy.

Your file should not contain any `print` and `echo` statements. Instead, insert dynamic content into the page using PHP expression blocks with `<?=` ... `?>` as taught in class.

Another grading focus is **redundancy**. You should not have code that depends on particular movies or uses `if/else` statements to see which movie to display. Use loops, variables, `if/else` factoring, etc. to avoid redundancy.

Another grading focus is PHP **commenting**. We expect more comments here, similar to CSE 14x. Put a descriptive header (name, course, TA, description) at the top of your code and comment each non-trivial section of PHP code explaining the purpose of that code.

Format your HTML and PHP code similarly to the examples from class. Properly use whitespace and indentation. Do not place more than one block element on a line or begin a block element past the 100th character. **CSS** is not a major part of this assignment, but your CSS file should validate and you should not introduce new bad/invalid CSS code.

Part of your grade will also come from successfully uploading your files to the **Webster** web server in a subdirectory named `hw3`, so that it is possible to navigate to your page by entering the following URL into the browser:

- **https://webster.cs.washington.edu/*your_uwnetid*/hw3/movie.php**

Please do not place a solution to this assignment online on a publicly accessible web site.