

CS306 Spring 2009

Bonus Homework Assignment (#6)

Due: Monday, May 4<sup>th</sup>, 2009 at 4pm

150 points (see note about grading)

**Grading.** This homework is optional. If you choose to skip it, your homework grade will be calculated out of the 750 possible points from HW1-HW5. If you choose to do submit a Bonus HW, then your HW grade will be graded out of 900 possible points. I will only include your Bonus HW in your HW grade calculation if it raises your grade, so this is a “no-risk” homework if you choose to do it.

### Program 1 (150 points). iPod

Your task is to design and implement an OO model of an iPod and its music library. All that I provide here are the class names and the API which your model must support. You are free to implement the internals of your class as you see fit.

Class: iPod (this implies an iPod.pm file) - This class represents an iPod music player. It has the following API:

- `new(name => $name, capacity => $capacity, battery_life => $minutes)`  
All of the parameters are required for this constructor. `name` is a string representing the name of this iPod. `capacity` is an integer, e.g. “8000000000”, representing bytes. `battery_life` is the number of minutes of battery life remaining, e.g. “482”.
- `$track_id = add_track(track => $track, before => $id)`  
`$track` is an object of type `MediaFile`. Returns a unique ID integer which identifies this track to this iPod. The optional parameter “before” specifies a `track_id` such that the new track should be inserted before that existing `track_id` (in effect, the new track should become the `$id` track, and all old tracks starting at position `$id` are bumped to position `$id+1`). If “before” is omitted, the new track is appended to the end of the track list. If the iPod does not contain sufficient space for the new track, an error should be displayed.
- `remove_track(track_id => $track_id)`  
Removes the track that has the `id` `$track_id` from the iPod.
- `@playlist = shuffle(length => $length, type => $type)`  
Returns a playlist of length `$length`, where each element is an object of type `MediaFile`, with no repeats, in random order (sequential calls should not result in the same playlist under most circumstances). Optionally, the user can specify `$type`, which is a class name matching one of the valid `MediaFile` subclass names. If `$type` is provided, the return list will only contain objects of class `$type` (e.g. “Podcast”). If `$length` is greater than the number of tracks (or the number of tracks that meet the `$type` filter) an error should be displayed.
- `list_tracks()`  
Shows the track listing, ordered by ID number, including the ID number assigned by the iPod and all of the track details, including details specific to a certain type of track (e.g. the episode number of a Podcast).

- `get_battery()`
- `set_battery(battery_life => $minutes)`  
Getter and setter for the `battery_life` attribute, where `$minutes` is an integer representing number of minutes of battery life remaining (e.g. 217).
- `$count = get_track_count()`  
Returns the number of tracks currently on the iPod.
- `$used = get_used_space()`  
Returns the number of bytes currently used on the iPod.
- `$remaining = get_remaining_space()`  
Returns the number of bytes currently available on the iPod.
- `print()`  
Displays information about this iPod, including its name, `battery_life`, capacity, space used, space remaining, and number of tracks.

Class: `MediaFile` – This is the parent class for all media files that can go on the iPod. It has the following API:

- `new()`  
A default constructor.
- `new(duration => $duration, title => $title, artist => $artist, bitrate => $bitrate, size => $size)`  
This constructor takes values for all of the `MediaFile` attributes.
- `get_duration()`
- `set_duration(duration => $duration)`  
Getter and setter for the duration attribute, where `$duration` is a `HH:MM:SS` formatted string.
- `get_title()`
- `set_title(title => $title)`  
Getter and setter for the title attribute, where `$title` is a string.
- `get_artist()`
- `set_artist(artist => $artist)`  
Getter and setter for the artist attribute, where `$artist` is a string.
- `get_bitrate()`
- `set_bitrate(bitrate => $bitrate)`  
Getter and setter for the title attribute, where `$bitrate` is an integer, e.g. "192", representing a bit rate of 192k.
- `get_size()`
- `set_size(size => $size)`  
Getter and setter for the file size, where `$size` is the number of bytes, e.g. "5223933".
- `print()`  
This is the parental print method, which displays the details of this `MediaFile` (the attributes that are common to all media files).

Class: `Song`

- `new()`  
A default constructor.
- `new(duration => $duration, title => $title,  
artist => $artist, bitrate => $bitrate,  
size => $size , album => $album)`  
A constructor which provides values for all attributes of a Song.
- `get_album()`
- `set_album(album => $album)`  
Getter and setter for album attribute, where `$album` is a string.
- `print()`  
This displays all of the details of a Song object, including the attributes which are stored here and the attributes that are stored in the parent class.

#### Class: Video

- `new()`  
A default constructor.
- `new(duration => $duration, title => $title,  
artist => $artist, bitrate => $bitrate,  
size => $size, aspect => $aspect)`  
A constructor which provides values for all attributes of a Video.
- `get_aspect()`
- `set_aspect(aspect => $aspect)`  
Getter and setter for aspect attribute, where `$aspect` is a string representing the aspect ratio, e.g. "16:9".
- `print()`  
This displays all of the details of a Video object, including the attributes which are stored here and the attributes that are stored in the parent class.

#### Class: Podcast

- `new()`  
A default constructor.
- `new(duration => $duration, title => $title,  
artist => $artist, bitrate => $bitrate,  
size => $size, episode => $episode)`  
A constructor which provides values for all attributes of a Podcast.
- `get_episode()`
- `set_episode($episode)`  
Getter and setter for episode attribute, where `$episode` is a string representing the episode number, e.g. "Vol 2, Episode 7"
- `print()`  
This displays all of the details of a Podcast object, including the

attributes which are stored here and the attributes that are stored in the parent class.

You will want to write a driver program in order to test your model. Although I am not providing my driver program, you can imagine that it will create an iPod, add tracks to your iPod (checking if your iPod correctly detects when it is full), print a track listing, generate a few playlists (checking to see if your iPod recognizes when I ask for too many tracks in my shuffle playlist), and then remove some tracks from your iPod.