

A Comprehensive Course List for Moodle by Using a Tree Structure

Weerarathna U.I.M., *Member, IEEE*
 Computer Science and Engineering,
 University of Moratuwa, Sri Lanka.
 E-mail: uisurumadushanka89@gmail.com

Abstract—In the Open-Source community the most popular learning management system is the Moodle. It is used by most of institutes in present, because it is open to modification as desired. Numbers of difficulties are faced by Moodle users due to lack of growth of its course list. Course list is a block which is responsible for showing subjects that are taken by a user. Issue is rising with the increasing of number of courses in the course list, because user is usually take more time to select relevant subject from list which is not categorized at all. This effort has used to reduce that difficulty by categorizing the courses using a tree structure.

Index Terms— Course list, Moodle, Moodle block, Tree List

I. INTRODUCTION

THE world is truly globalized with the improvement of the technology. Every time when we talk about technology, immediately the thing that comes to our mind is the Internet. Internet has influenced in most of our day-to-day lives of humans. It actually has become a human need. Institutes, universities or even schools are always looking to improve their productivity. So, the technology is used to their activities too. One area of such network technology is used, is managing courses and their contents which provided to the students. A system has to be used here and those are available at different levels on demand. Most popular learning management system in Open-Source Community is the Moodle. The reasons to increase the popularity of the Moodle are, it is easy to use and it is available to modify as users they want, hence improves the flexibility of the system according to the various cultures of users.

One of an area that most Moodle developers are not looking to improve is its course list. Course list shows all the subjects taken by a user. It doesn't categorize the courses at all. What it does is, it sorts the subjects according to alphabetic order of subjects' name and represents as a list. This makes a huge issue of selecting a course when the number of items in the list is so large. Our eye may not able to capture the necessary subject at once, making some time to familiar with the list and choosing it.

In this paper, I will describe how I achieved a solution to

this problem using a proper structure and methods. This paper has organized as follows. In the next section I introduce what technology is used to solve this problem. Then I will elaborate what approach was taken by me to understand user requirements. In section IV, I will explain how the implementation is done according to most programming standards.

II. MOODLE AND ITS TECHNOLOGY

▪ Moodle

If you already have used the Moodle, you may know or not that it is programmed using the PHP. Yes, it's true that it must have used some Open-Source scripting language for its development. Also, almost all dynamic page creations are done in the server side. Client side is only just showing the pages what it receives. In other words Moodle is built on 3-tier architecture.

All the data about users and their entities are stored in a database. Usually Moodle supports many database platforms but most frequently used database server is MySQL. Moodle's architecture is simple. Some of you might afraid by seeing thousands of files in Moodle, but the architecture is simple. It is simply a modular architecture, hardly any objects but lot of functions. Every entity in Moodle is a folder. Only thing to do to input a new entity to Moodle is create a folder or sometime a file according to the proper format and Moodle itself handles all automatic installation of those entities.

No other extra new technology has been used for this my new course list. If I used new technology here, all I have used has to be distributed with installation files. This may cause to incorrect operation of Moodle. So it has to be clearly and cleanly implemented the new course list. PHP has used for the generation of the block while JavaScript is used to control the behavior of the tree structure.

▪ Block Structure

It is better to explain how the block structure in Moodle is operating. All the blocks are stored in the 'blocks' folder in Moodle root directory. For each block there must have a folder and that its name should not be included any spaces or invalid characters, because Moodle parses its name as a variable and uses globally to identify it. If you want to separate words, use underscore (_) letter.

As the content of the block, you can create a file or several files implanting your functions. More important thing here is all your implementation must encapsulate within a class name `'block_[your_block_folder_name]'`. And also it must inherit from the class **block_base** or any sub class which inherit from block_base class. In there you have to override `'function get_content()'` method which returns the content of the block to be displayed. If you want to provide settings for your block, a settings.php file can be implemented in the block folder. This is optional.

Block types are controlled by `'moodleblock.class.php'` file. It includes concrete classes for block implementation. In here it can create new class types too.

All the operations regarding on blocks are done by `'blocklib.php'` file in the `'lib'` folder in Moodle's root directory. It contains all the functions like printing block headers, contents and footers, controlling behavior of block etc. It is not mandatory to use `'blocklib.php'` class to your block. If anyone like to have different functions than there exists, can be created a separate file and call those functions via `'moodleblock.class.php'` file.

III. DEMAND

Current course list has some issues regarding when increase of number of courses in the list. Users like to have clean and clear course list on their view to grab courses quickly. But this does not meet with current course list. Users like to see completed subjects to be hidden – if they want to see them and provide that too - and save space on the block. Also it is desire to categorize courses according to the level it belongs to and it makes clear cut of selecting courses quickly.

While providing those features we can't forget basic user requirements that Moodle users are expecting. Like simplicity, clearness, performance etc. All those have to be addressed by the solution.

IV. IMPLEMENTATION

Before any implementation is to be started I searched for a tree structure which I can use in my course list. First I thought about external tree structure which will be available in internet. But later I found out that there is already an internal tree structure which is using for administrator option displaying. (The tree list in the left hand side of the main page of administrator logged-in) More observations has done by me especially about where its origin and how the implementation of it has been done. This admin tree is located same directory in `'blocks'`. But the implementation is not generic, that is it is not reusable for other things due to the hardcode of nodes and all other things. So I had to look on another way to reuse it in my project.

Idea was raised in my mind to create a new tree list block type which may helps to other developers too in their future developments. So, I examined how to create a new block type. To create a new block type it has to use `'moodleblock.class.php'` file. Inside it we can create a new class with new parameters which may help to build a tree list

block. So creating it was my first step. At the beginning I didn't include any specific function calls inside overridden methods but more common ones. Then I began to implement course list block.

As the structure I had to use only arrays because in PHP it is hard and complex to implement any advanced data structures like trees. I decided not to use a root node for my tree list because in this scenario it is not important and it may take another level of nodes. Also I decided to limit the tree list only to two levels and no more. Reason for that decision is if I used more than two levels the horizontal space may not enough to display a tree node at deepest levels. Also this may increase the complexity of implementation. Following has described how the tree structure is made of.

- ❖ Each node is a new stdClass in PHP.
- ❖ All those nodes are stored in an array.

I used the advantage of using a stdClass for store node information in PHP. As you know in stdClass you don't want to implement explicitly each attribute and its type you want at all. You can use attributes in time as you want by calling to it assuming it is in there. PHP is responsible to create attribute automatically if the called attribute is not exist, otherwise returning the value. So, my tree node attribute as follows.

Attribute	Description
id*	Stores node identification number.
title*	Display text of node.
items[]*	Array indicating children items of node. Just only by text.
icons[]*	Array indication icon for each children node.
visibility[]*	Array storing the visibility of each children node.
flags[]	Array of storing extra information on processing each node.
recent_activities[]	Array of storing number of recent activities for each course.
footer	Footer text for node.

Node structure in tree list

* - mandatory when implementing your own tree list.

In the `block_course_list.php` file I have assigned values for each of attributes above mentioned by taking every course which user is currently taking. To decide whether a course is completed or not I have to use extra strategy because that kind of implementation hasn't been implemented so far. I observed in the database there is an attribute number of weeks and starting date for each course record. That was useful in implementing a function to identify the course end time.

When starting to implement that function I had to think twice in where should I implement it? I could have easily used blocklib.php module in the `'lib'` directory on the Moodle's root directory. But as a good programmer it is better to not to hardcode in these open-source environments. Hard coding can cause damage to existing functionality of Moodle when the module is distributing. Especially in case of the target user has

already modified its module as they want and replacing that same module with ours may not have those changed functionalities at all. By all those reasons led me to create a separate library module. It also may helpful even when installing process.

Now the important part has to be implemented in that library module that is printing function of the tree block. We can't use currently existing printing function or change it because it supports only text type and list type blocks. Since tree functions are more different than above those two types I had to create a new function for printing. As usual there should have three printing functions one is for block header printing, one for content printing and the other is for footer printing. In my case, header printing is same as other blocks but content and footer is different.

Content printing is the most complex part. Because PHP doesn't have in-built sorting functions which helpful in my case to filter courses according to completeness. All things have to be implemented by myself very carefully with having consistent code too. In sorting currently on-going subjects have to be filtered on top of the node items while completed ones are at the bottom. Following indicates the pseudo code for sorting method.

```
for each (node in tree)
  while (not all items in node are added)
    for each (item in node)
      if (item has on-going)
        print node item
        mark the node as added
```







Pseudocode for sorting function.

There is one thing I included here that is I have also used cookie data to store and retrieve node's status whether it is expanded or collapsed already. This is very helpful in users' side because user may not want always to spend his/her time on expanding/collapsing nodes. The status of nodes is already being stored in the cookies.

At the footer printing I have to input the behavior of the tree list in JavaScript. Those behaviors are added to the html document by scripting as java script inside a Meta tags.

■ Images

Following images have been used for the project and tree list.

Image	Image File Name	Description
	expandall.gif	Used for settings icons
	collapseall.gif	
	tick_green_big.gif	Completed course icon
	arrow_collapsed.gif	Node status icons
	arrow_expanded.gif	
	course.gif	General course icon

■ Module summary

Module	Description	Functions /Lines
coursetreelib.php	Includes all the functions required	9/1110

	for course tree list.	
block_course_treelist.php	Includes all the strings that are used by block. Helps for localization.	0/49
moodleblock.class.php	Includes base class for course tree list.	3/90
block_course_list	Includes base implementation of course information.	5/375
settings.php	Includes settings which may require in administrator side.	0/52
Total Lines of code approximately = 1676		

V. FLEXIBILITY

It is true that every user's desires are not equally distributed. Various users have various desires. Satisfying all those requirements is impossible. But basically we can do fair adjustments to our projects which may satisfy most customers. To do such thing we can provide some settings adjustment manually by users. In this project I have implemented two sided settings types, one is administrator side and other is personal side. Administrator side settings are provided via 'settings.php' file in 'block_course_list' folder. Following indicates the brief description about those settings.

Setting	Description
Type of categorization	Type of categorization whether nodes are sorted according to completeness or by levels.
Show total activities	Show/hide number of recent activities since user's last log-on.
Completed Font Size	Which font-size to be used for completed subjects.
Hide node for completed courses	Automatically hides nodes which all courses are completed.
Collapse all completed nodes	Automatically collapse nodes which all courses are completed.
Different icon for completed	Use distinguishable icons for differentiate completed and on-going courses.
Show settings in course block	Shows/hides the personal settings for course block.
Start level of postgraduates	The level which postgraduate courses are started.

These administrator settings are displayed under **Modules > Blocks > Course** in administrator main page. Those settings are never accessible by general users. So Moodle administrator should aware the best settings which suitable for their institute.

Other type of setting is Personal settings which allow individual users to customize the behavior according to their personal desires. This is more limited section than administrator settings, but assumes more powerful. These settings are shown just above the course tree list contents in the block. There are one check box and two links. Check box used to quickly show/hide the completed courses in tree list. When the number of courses are increasing, the unnecessary courses – completed courses – may not want to display or sometimes users may want to access to completed courses. To provide this ability it is poor design to implement this setting on administrator side. Other two links are used to quickly expand and collapse tree nodes. This is extremely important in time saving by reducing some clicks. More important thing regarding on these personal settings is it has the ability to save your personalized settings even you logged-out by using cookies. In next time you logged-in to the system those saved settings will be automatically loaded. Not only settings but also the node status – like it is expanded or not – are capable of saving and restoring. This ability is so much valuable in user's side because user may not want to do same adjustments again and again. There is one thing to remember that those settings are saved only for 30 days period. That is if you didn't log in to system for a 30 day period, your settings will be expired and default settings may take in place. If you want to change the length of time taken to store settings can be increased via the code by changing only value in a variable.

VI. CONCLUSION

In this paper I proposed a comprehensive course list has been implemented by using a tree list for Moodle users to easier their course picking. In future I hope to extend this project into which the recent activities are shown in tool tips when you move your cursor on the course item. Then there won't be any necessary to click the course and search for where are the recent updates.

ACKNOWLEDGMENT

Special thanks goes to my parents for helping throughout this project even they don't know nothing about computer but providing recourses when I am in trouble. Also thanks goes to our level 3 project advisors and coordinator Ms. Shahani Markus Weerawarne who taught us more important concepts and principles about software engineering and its field. And also all the other staff who helps me throughout this project development period. They all are warmly acknowledged.

REFERENCES

- [Online]: <http://docs.moodle.org/en/Development> : Developer Documentation of Moodle, 6th of March 2011.
- [Online]: http://www.w3schools.com/PHP/php_cookies.asp : PHP Cookies, 6th of March 2011.
- [Online]: <http://www.w3schools.com/php/default.asp> : PHP, 6th March 2011.