Towards Adapting Sakai for e-Learning Provider

Jyoti Chauhan¹, Khulan Batbayar¹, Richa Sharma¹, Deepika Sharma¹, Divya Popli¹, Naveen Kumar¹ and Anita Goel²

¹Department of Computer Science, University of Delhi, New Delhi, India
²Department of Computer Science, Dyal Singh College, University of Delhi, Lodhi Road Delhi, India

Keywords: e-Learning, Educational Software, MOOC, Sakai Skin Changing, SkinChanger.

Abstract: Massive Open Online Course (MOOC) is an emerging method in online education. Sakai is a popular open source platform widely used by e-learning providers to offer MOOC. Sakai provides a set of collaborative tools for teachers, learners and researchers to support teaching, learning and research. For using Sakai, the e-learning provider needs to customize and adapt the interface of Sakai, like, banner, logo, background color etc. to suit their requirement. This paper presents the development of a tool named SkinChanger that facilitates customizing of Sakai interface, based on need of the e-learning provider. SkinChanger customizes the appearance of Sakai body, header and footer. It facilitates skin changing by providing a user friendly interface. SkinChanger is component-based tool and can be plug-in. This paper illustrates the use of tool to customize the Sakai interface for University of Delhi.

1 INTRODUCTION

e-Learning is impacted through the development of technology. Massive Open Online Course (MOOC) is an emerging method in online education that came into existence in 2008. It promotes formal and informal education via open enrolment, unlimited participation and allows access via the web (Wikipedia, 2014). There are several MOOC providers that support e-learning, like, Coursera, Udacity and edX.

The e-learning providers can set up their own new MOOC platform or adapt an existing one. Generally, an open-source platform is customized by the e-learning provider to suit their needs. Some of the available open-source MOOC platforms are Open edX by edX, CourseSites by Blackboard, Versal, CourseBuilder by Google and Sakai. The Open edX requires investing in setup and maintenance; CourseSites limits the number of courses up to five; Versal lacks discussion forum, a key feature of MOOC (Swope, 2014). Sakai is a popular platform having a community of developers from all over the world supporting it.

Sakai is an open-source educational software platform used by e-learning providers to offer MOOC. It provides Collaborative and Learning Environment (CLE) and it is aimed to develop a set of collaborative tools for teachers, students and researchers that supports teaching, learning and research (Alves et al., 2012, p. 50). Sakai is used by more than 350 world’s great colleges, universities and organizations of diverse profiles list; over 4 million learners worldwide (Sakaiproject.org, 2014) which shows it’s acceptance in education industry.

The Sakai Project follows Model View Controller (MVC) architecture for programming. It provides a modular architecture in which various tools, services and resources are combined within a single, access-controlled framework (Fraser, 2005 cited in (Rimpiläinen and Carmichael, 2006, p.2)). Thus, a new feature can be easily incorporated in it.

Sakai interface comes with a default look and feel. Any e-learning provider having interest in adopting Sakai, need to customize its appearance, like, logo and banner of their organization. The provider hosting Sakai needs to configure Sakai’s skin for making these changes.

In Sakai, there are three popular ways of customizing the skin - (1) choose from pre-installed skins in Sakai package, (2) manually setup up a new skin, and (3) use a tool for customization of skin. When using the first two options for skin changing, there is a need to understands the Sakai file structure and skin installation process. SkinManager (Knoop
and Groen, 2007) is an existing tool developed by Edia Educatie Technologies to automate the process of skin customization in Sakai. Although the SkinManager allows provider to choose from the listed archived skins, but, to define a new skin, the provider need to have prior knowledge of style sheets, Sakai file structure, storage directories and graphics. The using of existing ways to customize the skin of Sakai does not provide flexibility and is a tedious task.

In this paper, we focus on development of tool that facilitates the Sakai adapters to customize its appearance. The tool promotes flexibility by allowing the user to customize the Sakai skin as per their requirement, via an interactive interface. In the rest of the paper, we use the term user for the organization or the provider using Sakai for e-learning. Using our tool, the user can change skin by selecting desired options in contrast to writing fresh code for the same. Our tool focuses on customization of existing text properties like, color and font style; rather than text editing and reordering of elements (menu, table, calendar, button).

Here, we present SkinChanger, a tool for customization of skin of Sakai for e-learning providers. It is composed of three main components- (1) Body-Skin, (2) Header-Skin and (3) Footer-Skin. Body-Skin component allows style changes in body of the displayed gateway screen. Likewise, changes in top portion of the screen are done using the Header-Skin and changes required, like, copyright styles that are located in the bottom portion are performed by Footer-Skin component. Each component is further divided into sub-components where their corresponding properties to revise the skin are defined.

The tool is developed as a pluggable component. It needs to be installed before running the instance of Sakai. Using the tool does not require any particular specialization or expertise.

The SkinChanger tool is implemented in Java and developed using Sakai Wicket maven Archetype packages based on Apache Wicket. The package allows generation of a sample Sakai app. The app is based on Apache Wicket which integrates nicely with Sakai (Swinsburg, 2014). Using the Wicket does away with the need of XML configuration files. The versions used in the development of tool are JDK 1.7, Sakai 2.9.x, Apache Tomcat 7.0.52, and Maven 3.1.1.

In this paper, section 2 is a survey of related work. Section 3 presents the interface of Sakai. Section 4 explains the SkinChanger and its features. Section 5, 6 and 7 discuss the main components of the tools - body, header and footer, respectively. Section 8 enumerates use and implementation details of SkinChanger. Section 9 states the conclusion.

### 2 RELATED WORK

Several researches exist in area of customization of interfaces. Some are specific for software like, Word Perfect (Page, Albert and Allen, 1996), Microsoft Word (McGrenere, Baeccker and Booth, 2007), and DISCIPLE (Dorohonceanu, Sletterink and Marsic, 2000). Others focus on customizing interfaces for a specific purpose like, simulation (Nichols and Stanton, 1990), multi-user interface (Rodham and Olsen, 1994), GIS (Lopes de Oliveira, Mederiso and Cilia, 1997), online travel planning (Jurca, 2000) and supporting group collaboration (Dorohonceanu, Sletterink and Marsic, 2000). The customization is also targeted for a special class of users, for example, blind (Winberg, and Hellstrom, 2003) and different skill level (MacLean et al., 1990) users. Cognitive characteristics and psychological behaviour of the user is also used to adjust look and feel of the interface for individual or group of users (Benyon, 1993; Benyon and Murray, 1993).

Adaptive systems are used for providing the customized services to the user that supports automatic customization of the user interfaces (Cockburn, Gutwin and Greenberg, 2007; Findlater et al., 2009; Gajos et al., 2006; Greenberg and Witten, 1985). But adaptive approach mainly changes presentation of content rather than interface elements. Advance options for customization are also provided in (Stuerzlinger et al., 2006; C. Zeidler, Lutteroth and Weber, 2013). Eigo Okada et al. (2008) propose a framework for customization of menu structure and accessibility of functions using hyperlinks.

The manual process of customization of Sakai provides best results as by programming a fresh code almost any element of interface can be customized. But, it can be done only by an expert having knowledge of programming Sakai.

Generally, use of tool reduces need of domain knowledge and programming skills. The already existing SkinManager (Knoop and Groen, 2007) tool allows to change appearance of Sakai by choosing from archived skin or defining a new skin. Both options have their own limitations. When selecting a pre-defined skin, it has predefined logo, banner etc. which might not be useful for organisation adapting Sakai. For defining a new skin there is a need to have the knowledge of CSS programming and expertise in Sakai.
Table 1: Evaluation of customization methods of Sakai.

<table>
<thead>
<tr>
<th>Features</th>
<th>Manual</th>
<th>Existing Tool–SkinManager</th>
<th>Proposed Tool–SkinChanger</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Process</strong></td>
<td>Locate files that needs to be changed</td>
<td>Select a skin from archive or Upload a new skin</td>
<td>Select component and corresponding properties</td>
</tr>
<tr>
<td>Program CSS code for change Configure file to load changed skin</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Process Automation</strong></td>
<td>–</td>
<td>Only for manual process’s configure file step</td>
<td>For all steps of manual process</td>
</tr>
<tr>
<td><strong>Technical Expertise</strong></td>
<td>CSS programming, Sakai file structure, storage directory</td>
<td>CSS, Sakai file structure, and storage directory knowledge – to define new skin</td>
<td>Not required</td>
</tr>
<tr>
<td><strong>Scope</strong></td>
<td>Customization flexibility up to full extent, need extra efforts and expertise</td>
<td>Select a skin – provide skins with predefined parameters that might not be in useful for another organization Defining new skin need expertise</td>
<td>Allows customization based on user’s selection from the listed component, subcomponent and properties</td>
</tr>
<tr>
<td>Partial customization</td>
<td>Possible – write specific new code</td>
<td>Not possible</td>
<td>Possible – Select specific component and its parameters</td>
</tr>
</tbody>
</table>

Table 1 summarises evaluation of customization method for Sakai.

To best of our knowledge, we could not find any work that provides skin customization of Sakai based on the user’s selection. SkinChanger, our tool, does not require any prior knowledge for changing.

3 SAKAI INTERFACE

The Sakai provides a user-friendly interface for learners and educational providers. It provides easier navigation via menus. The logo, banner, tagline of organization, information about the course etc. are displayed in the header of page. The available sites, enrolled students for each site, type of role assigned for a site to student, available resources etc. are displayed in the form of table. The events and schedules are highlighted in calendar. To get details regarding a particular course site, the information is displayed in a tabular form. The footer of the page displays the quick links and copyright information.

Figure 1 shows the interface of Sakai with its default skin. It includes banner, header color, menu, links, and copyright information. In figure, some sections and properties that need to be customized by the adapters of Sakai are highlighted. For example, logo, banner, header color, calendar style, table style, copyright information, quick links, font color and background color. The e-learning providers adopting Sakai need to customize the look and feel of the skin of Sakai.

Figure 1: Default Sakai 2.9.x interface with highlighted banner, header color, menu, link and copyright.

Figure 2: Sakai page layout division in header, body of page and footer.

4 SKINCHANGER: A TOOL

SkinChanger is a tool developed to provide an interface for changing the look and feel of Sakai according to requirement of e-learning provider. Page layout of Sakai is divided into sections as
shown in Figure 2. Components are designed for each section based on their role of customizing a specific portion of page. SkinChanger has three components as follows-

1) Body-Skin
2) Header-Skin
3) Footer-Skin

The Body-Skin component is responsible for formatting the body of page, like, menu, table, calendar and style of content. Customization of site navigation area and contents in top portion of page is done using the header-skin component. Footer-skin component allows making changes in the footer portion of page, like, copyright information.

The key features of SkinChanger tool are explained as follows:

Change Appearance of skin when an organisation needs a different look and feel for their Sakai portal, for example, logo, banner, background color etc. Using this interface they can change layout of body, header and footer of the page according to their needs.

Conditional Formatting is a feature that facilitates formatting of the skin based on selection of the component and its properties. The organisation may need to change only a portion of skin. SkinChanger enables customization based on requirement selection for the organisation.

Selection helps to make SkinChanger interface user friendly. By selecting a section or sub-part, the user will get the list of styles available for the selected component. For defining a new skin for Sakai, user only needs to select the component and its available style.

Independency - The SkinChanger is developed as a different component, so its inclusion does not affect any already existing tool or service of Sakai.

Plugin is a feature of SkinChanger that treats tool as an independent component that can be plugged-in whenever required.

The following sections discuss components of the SkinChanger tool.

5 THE BODY COMPONENT

The Body-Skin component is used to customize the main portion of the page. The component consists of four sub-components- a) Body_Menu, b) Table, c) Calendar and d) Body_Style. Table 2 summarizes the components of Body-Skin with their properties.

Body_Menu sub-component customizes the menu of page. It is divided into Menu_style, Menu_Selected_style and Menu_MouseOver_style.

Style of menu is customized using Menu_style. Menu_Selected_style is used to change menu style when a user selects it. Menu_MouseOver_style is responsible for changing menu styles whenever mouse pointer scrolled over it. Each of these styles provides list of style properties based on the selected style. A user can select some, all or no properties. Figure 3(i) illustrates customized menu with color change and mouse over option.

Table sub-component defines formatting style for activities listed in the table. It consists of Table_Header_Design and Table_Content_Design. To customize the header of table along with its properties, like, width of table border, font weight etc., Table_Header_Design is used. Table_Content_Design is assigned the task of making changes in properties for content of the table, like, text vertical alignment, border color of top & bottom, bottom size of top & bottom, and highlight added item style of table header. Figure 3(ii) shows customized table reflecting changes in header color.

Calendar sub-component customizes the listed events belonging to a site or user. Calendar is divided into Cal_Header_Style and Cal_Day_Style. Cal_Header_Style helps to modify header information of the calendar. For example, changing background color of calendar header and define parameters of border. Customization of the date displayed in calendar is under Cal_Day_Style. Based on different types of day styles it is further classified into Cal_Today, Cal_Activity, and Cal_NoActivity, to customize the current date, associated date with an event and date not associated with any event, respectively. Each of these day styles have properties like, border width, text alignment. Figure 3(iii) shows customized calendar of changed background color of date associated with an activity.

Figure 3: Screenshots of customized Body skin (i) Body_Menu, (ii) Table, and (iii) Calendar.
Table 2: Body-Skin.

<table>
<thead>
<tr>
<th>Sub-components</th>
<th>Classification</th>
<th>Properties</th>
</tr>
</thead>
<tbody>
<tr>
<td>Body_Menu</td>
<td>Menu_Style</td>
<td>Text (Color, Alignment) Background color, Border style</td>
</tr>
<tr>
<td></td>
<td>Menu_Selected_Style</td>
<td>Text color, Background color</td>
</tr>
<tr>
<td></td>
<td>Menu_MouseOver_Style</td>
<td>Text (Color, Decoration), Background color</td>
</tr>
<tr>
<td>Table</td>
<td>Table_Header_Design</td>
<td>Border (Width, Style), Font weight, Background Color</td>
</tr>
<tr>
<td></td>
<td>Table_Content_Design</td>
<td>Vertical alignment, Border Color (Top, Bottom), Border Size (Top, Bottom)</td>
</tr>
<tr>
<td>Calendar</td>
<td>Cal_Header_Style</td>
<td>Background color, Border (Width, Style)</td>
</tr>
<tr>
<td></td>
<td>Cal_Today</td>
<td>Border (Width, color)</td>
</tr>
<tr>
<td></td>
<td>Cal_Activity</td>
<td>Background color, Text alignment</td>
</tr>
<tr>
<td></td>
<td>Cal_NoActivity</td>
<td>Background color, Text alignment</td>
</tr>
<tr>
<td>Body_Style</td>
<td>Horizontal_Rule</td>
<td>Color, Width, Border style</td>
</tr>
<tr>
<td></td>
<td>Link_Style</td>
<td>Text color, Link Text decoration, Link_MouseOver (Text color, Decoration)</td>
</tr>
<tr>
<td></td>
<td>Text_Heading</td>
<td>Color</td>
</tr>
</tbody>
</table>

Table 3: Header-Skin.

<table>
<thead>
<tr>
<th>Sub-components</th>
<th>Classification</th>
<th>Properties</th>
</tr>
</thead>
<tbody>
<tr>
<td>Header_Style</td>
<td>Configuration</td>
<td>Background color, Logo image file, Image (Width, Height)</td>
</tr>
<tr>
<td></td>
<td>Before_Login</td>
<td>Height, Banner image, Banner image (Width, Height)</td>
</tr>
<tr>
<td></td>
<td>After_Login</td>
<td>Height, Banner image, Banner image (Width, Height)</td>
</tr>
<tr>
<td>Header_Menu</td>
<td>Style</td>
<td>Background color, Font weight</td>
</tr>
<tr>
<td></td>
<td>MouseOver_Style</td>
<td>Background color, Border</td>
</tr>
<tr>
<td></td>
<td>Selected_Style</td>
<td>Background color, Selected border</td>
</tr>
</tbody>
</table>

Figure 4: Screenshot of customized header (i) before & after login, (ii) menu, and (iii) header.

Body style sub-component handles styling of the page body. The task of customization of the body of page is done using Link Style, Horizontal rule, and Text heading. Page has different types of links, like, visited, non-visited, and active that helps to navigate to some other page. Horizontal rule is assigned to change color, width and border style of these lines. Text Heading allows changing color of the text.

6 THE HEADER COMPONENT

Customization of top portion of the page is done using Header-Skin component. The functionality provided by the header is divided into two sub-components Header_Style and Header_Menu. Table 3 presents header components and their properties. Header Style sub-component customizes content in the header of page, like, logo, banner and label. The task of formatting the Header style is further classified into Configuration, Before Login and After Login. The properties like, logo image file, image parameters are modified by Configuration. Header style is different when user login and it changes when a user logs out. Before Login and After Login are assigned to customize the header.
before and after log in, respectively. The properties like, logo image file, image parameters are modified by Configuration. Both before and after login style provides option to customize height, banner image, and banner image parameters (width, height) of header. Customized header of the gateway before and after login is shown in Figure 4(i).

Header Menu allows customization of information, like, my workspace, course and project names. It is further divided into Style, MouseOver_style and Selected_Style. Style is responsible for formatting of the menu, for example, change background color and font weight of menu. MouseOver_style customizes menu when mouse pointer scrolls over it. Customization of selected menu is done using Selected_Style. It facilitates the user to change properties, like, background color, border of menu. Figure 4(ii) shows changed background color of header menu. Customized header after changing its header style and its menu is shown in Figure 4(iii).

7 THE FOOTER COMPONENT

The information like copyright, quick links, server information, and version information is customized using Footer_Style sub-component. It is categorized into Link_Style and Copyright_Style. Table 4 lists the footer component with their properties.

<table>
<thead>
<tr>
<th>Sub-components</th>
<th>Classification</th>
<th>Properties</th>
</tr>
</thead>
<tbody>
<tr>
<td>Footer_Style</td>
<td>Link_Style, Copyright_Style</td>
<td>Text (Color, Decoration), Element space (Top, Bottom) Text color, Background color</td>
</tr>
</tbody>
</table>

Link_Style customizes the links available in the footer for navigation. The formatting of copyright information is under Copyright_Style. Each provides the option to change their properties. Figure 5 shows the customized copyright information and quick link in the footer.

8 USING THE TOOL AND ITS IMPLEMENTATION

Whenever an organisation needs to change the appearance of Sakai portal, the tool only needs to be plugged in. It makes the tool available in the list of available menus as shown in Figure 6(i). User can change a portion of page by selecting a section, subsection and properties. Figure 6(ii) depicts selected header component from the list and their subcomponents are shown in Figure 6(iii). Main components and their properties are as in Figure 7.

Here we illustrate use of SkinChanger for customizing the default gateway screen of Sakai 2.9.x. We have used it to customize the appearance for University of Delhi. Figure 8(i) shows default Sakai home page and customized home page with Delhi university logo, banner, color changes etc. is shown in Figure 8(ii).

Apache wicket and Eclipse have been used to develop SkinChanger that implements it as Eclipse plug-in. SkinChanger is then plugged into the Sakai portal. The component based model of Sakai supports inclusion of new tools or services.

Sakai follows Model View Controller (MVC) architecture for programming. Therefore, designing and implementation of the tool is done in three layers as depicted in Figure 9. Model provides data objects for resource accessing mechanisms. Data Access layer at physical layer use the DAO (Data Access Object) APIs and DAO Implementation to access and save data corresponding to skin. Controller uses logic APIs and its implementation for business logic layer. It retrieves the data from Model and provides it to presentation layer.

The presentation layer of the tool is designed in Java Server Faces (JSF) and it separates presentation from tool logic. The classes used for SkinChanger are separated according to the layered architecture of Sakai as shown in Figure 10. This tool focuses on presentation layer implemented in web pages using JSF and HTML. Web pages are in WebApps of the Sakai where separate HTML files are used for each section and subsection. Logic interacts with the Model using the APIs for accessing the resources.
The four main classes used by SkinChanger are BasePage, ColorPickerTextField, DataBase and FileHanding. The selected section and corresponding subsection provided by the interface are managed by DataBase class. FileHandling class is created for accessing files of the skin and exception handling, like, creating a file if it is not found or handle a generated exception. The BasePage class of Sakai extends by adding HomePage class of SkinChanger tool. HomePage class uses Database class and instantiates FileHandling public class. All the classes created for sections and subsections use the extended class ColorPickerTextField for rendering of color. Figure 11 shows UML diagram of Java classes used for the tool. In accordance with the wicket programming style, java files are created corresponding to each HTML file. SkinChanger uses the following classes:

- **Main page** - HomePage.html, HomePage.java
- **Panel associated with each style** - HeaderMainPanel.html, HeaderMainPanel.java, HeaderMenuPanel.html, HeaderMenuPanel.java, MainContentPanel.html, MainContentPanel.java, MainToolMenuPanel.html, MainToolMenuPanel.java, MainTablePanel.html, TablePanel.java, CalendarPanel.html, MainCalendarPanel.java, FooterLinkPanel.html, and FooterLinkPanel.java
- **Color chooser TextField component** – ColorPickerTextField.java
- **POJO(Model) classes** – Section.java, SubSection.java
- **File reading and writing** – FileHandling.java
In this paper, we present the adaptation of Sakai skin for e-learning providers. We present the development of a tool, SkinChanger that changes the appearance of Sakai CLE. The SkinChanger tool provides an interactive interface to educational providers for customizing look and feel of the Sakai.

The interface provides a menu based system in order to provide a user friendly environment. Furthermore, a user does not have to learn any sort of programming skills and expertise in Sakai file structure, to change the look and feel of Sakai. The user can just use the tool to select the element and their properties that have customization need to fit in the requirement of the organisation. For using the tool, it only requires to be plugged in before running the instance of Sakai environment. The non-requirement of any specific skill and degree of extent of customization makes it different from all the earlier work.

9 CONCLUSIONS

In this paper, we present the adaptation of Sakai skin for e-learning providers. We present the development of a tool, SkinChanger that changes the appearance of Sakai CLE. The SkinChanger tool provides an interactive interface to educational providers for customizing look and feel of the Sakai.

The interface provides a menu based system in order to provide a user friendly environment. Furthermore, a user does not have to learn any sort of programming skills and expertise in Sakai file structure, to change the look and feel of Sakai. The user can just use the tool to select the element and their properties that have customization need to fit in the requirement of the organisation. For using the tool, it only requires to be plugged in before running the instance of Sakai environment. The non-requirement of any specific skill and degree of extent of customization makes it different from all the earlier work.


