OULUN SEUDUN
AMMATTIKORKEAKOULU

Mikko Loimulahti

EVENT PASS GENERATOR

EVENT PASS GENERATOR

Mikko Loimulahti
Bachelor's Thesis
Spring 2012
Degree Programme in
Information Technology and Telecommunications
Oulu University of Applied Sciences

TIIVISTELMÄ

Oulun seudun ammattikorkeakoulu
Tietotekniikan koulutusohjelma, ohjelmoinnin suuntautumisvaihtoehto

---

Tekijä: Mikko Loimulahti
Opinnäytetyön nimi: Event Pass Generator
Työn ohjaaja: Eero Nousiainen
Työn valmistumislukukausi ja -vuosi: Kevät 2012
Sivuja: 35

Tämän opinnäytetyön tavoitteena oli tuottaa tapahtumapassigeneraattori, jolla voidaan tulostaa tapahtumakohtaisia henkilöllisyystodistuksia tietokantaan tallennettuja ilmoittautumisten tietoja käyttäen. Ohjelma tehdään osaksi suurempaa kokonaisuutta, Koululiikkuu Suomi – ilmoittautumisjärjestelmää. Asiakkaana ja toimeksiantajana toimi Koululiikuntaliitto, KLL.

Työssä käytetään iteratiivista ohjelmistonkehitystä, millä jatkuva palaute ja nopea muutoksiin reagoiminen on mahdollista. Toteutustapa myötäilee Koululiikkuu Suomen tapaa ja tapahtumapassigeneraattorista tehdään selaimella käytettävä sovellus.

Työ opetti tekemään käyttöliittymiä web-sovelluksiin, framework:ien tarpeellisuuden selainyhteensopivuuden takaamiseksi, sekä etsimään moderneja ja muodikkaita uusia nimiä jo vakiintuneille työtavoille.

Asiasanat:
tapahtumapassi, kisapassi, ilmoittautumisjärjestelmä, koululiikkuu, verkkosovellus

ABSTRACT

Oulu University of Applied Sciences
Degree Programme in Information Technology and Telecommunications

---

Author(s): Mikko Loimulahti
Title of thesis: Event Pass Generator
Supervisor(s): Eero Nousiainen
Term and year when the thesis was submitted: Spring 2012
Pages: 35

The aim of this thesis was to produce an event pass generator. It can be used to print ID cards for participants of various events. Participant data is fetched from a database to produce print-ready PDF files with personified info on each card. The work was part of a larger project, Koululiikkuu Suomi event registration system. The client for this work was Koululiikuntaliitto, KLL.

Since this product was integrated with Koululiikkuu Suomi, it uses same technologies and tools. It is a web application and software development was done using iterative and incremental methodology.

The work taught development of web application user interfaces, importance of frameworks for browser compatibility, and search and application of contemporary buzzwords when describing previously established working practices.

Keywords:
Id card, web, application, koululiikkuu, event, registration, pass

PREFACE

The work behind this report was done in summer 2011. This thesis focuses on using PHP and JavaScript frameworks to produce a database driven web application. I want to thank the supervisor for this thesis, Eero Nousiainen for his unrelenting encouragement, without which I could not have made this; and Elina Bergroth for examining my thesis.

TABLE OF CONTENTS

SYMBOLS AND ABBREVIATIONS

CSS	Cascading Style Sheet
DIV	A HTML tag used for page layout and styling.
DOM	Domain object model.
HTML	Hypertext Markup Language.
IDE	Integrated development environment.
MVC	Model-view-controller, an architectural pattern used in software engineering.
MySQL	Relational database management system.
PDF	Portable document format.
PHP	Nonsensical acronym for PHP programming language and interpreter.
SCRUM	Not an acronym but sometimes spelled with capital letters.
SQL	Structured query language.
TCPDF	Open source PHP class for generating PDF documents.
WYSIWYG	What-you-see-is-what-you-get.
XAMPP	Cross platform, Apache HTTP server, MySQL, PHP, Perl.

1 INTRODUCTION

Koululiikkuu Suomi –project is an event management system for publishing, reporting of and collecting entries for school exercise events. For a comprehensive event management system, involvement throughout the events' life cycles from planning to debriefing is a natural expectation.

One significant part of public events is personal identification. Events may have closed areas and 'invitation-only' parts for paid or selected individuals such as members of press, photographers, catering staff and athletes themselves. Entries may also want to order services or goods when they sign up for events, and delivery of such orders or purchases need identification at events.

Personal identification document for public events is usually called an event pass. Credit card sized, sometimes including a pass photo, it has printed information of, for example, personal contact details, registration data, appointed staff role, participation details and a list of ordered services and allowances.

Event passes are also mementos for participants and advertisements for events and their sponsors. Sometimes printed on special quality papers, passes may have logos, photos, images and other graphical content.

When all these various needs for an event pass are put together, an obvious conclusion can be drawn. An event pass generator must compete with professional graphic and desktop publishing software. Another option is to concentrate on the representation of participant information while leaving handling of graphical aspects for other programs. Applying this idea to the project, the practical solution was a making an upload function for readymade background images and superimposing participant information on those.

2 KOULULIIKKUU SUOMI -PROJECT

The target environment for the thesis was a web server with PHP and MySQL. These were chosen because Event Pass Generator was to be integrated into a larger project, Koululiikkuu Suomi, and thus it has to operate in the same environment.

Koululiikkuu Suomi is an event management system, which allows for advertising and on-line signing up for sports and exercising events of schools. It has been made for and owned by Koululiikuntaliitto, a Finnish association for student exercise.

Originally, named then as Live Project, Koululiikkuu Suomi was the result of multiple theses. Its database development provided enough scope for two theses and its user interface development was the subject of one more thesis.

3 SOFTWARE DEVELOPMENT PROCESS

Making computer programs professionally is done in an activity called software development process. A decades long goal of software development organizations has been to find repeatable and predicatble processes that improve productivity and quality. (1.) This chapter describes few theories of software development and which theories were used and how they were used in this project.

3.1 Theory

There are different methodologies for running a software development process. Modern methodologies are built on strengths of their predecessors. One modern methodology is Scrum and it is one in a group of several so-called lightweight software development methods. These methods are typically referred to as agile methodologies. (2.)

3.1.1 Agile software development

Agile software development gives a name for and quantifies methods that come naturally to people working in teams towards common goals. It attempts to sidestep heavyweight methods which were characterized by their critics as a heavily regulated, regimented, micromanaged, waterfall model of development. (2.)

> Agile manifesto:
>
> We are uncovering better ways of developing software by doing it and helping others do it. Through this work we have come to value:
>
> Individuals and interactions over processes and tools. Working software over comprehensive documentation. Customer collaboration over contract negotiation. Responding to change over following a plan.
>
> That is, while there is a value in the items on the right, we value the items on the left more. (2.)

Agile software development process is usually pictured with cyclical and incremental loops. (2) (Figure 1)

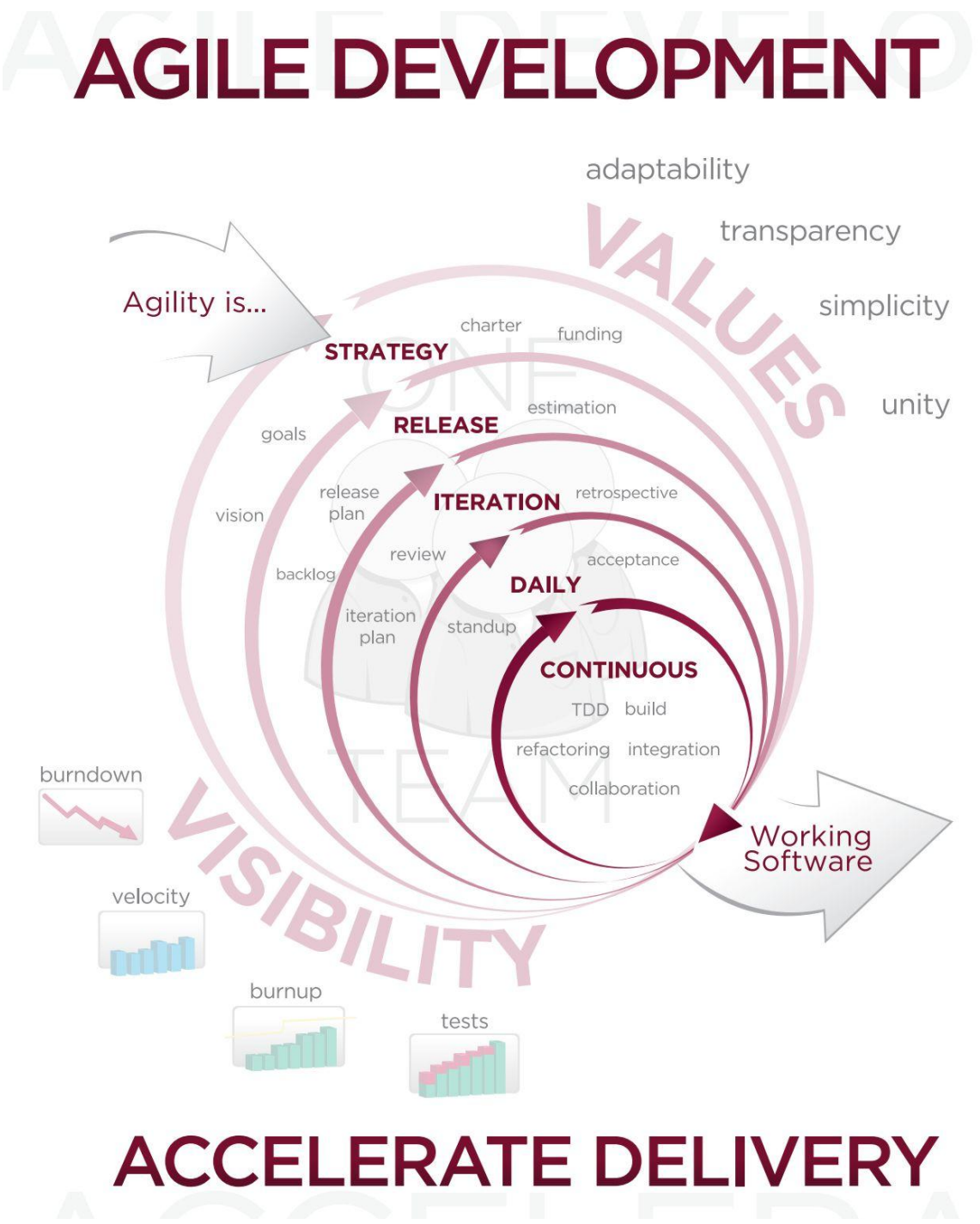

FIGURE 1. Agile Software Development Process (1)

3.1.2 SCRUM

Scrum is an iterative, incremental methodology for project management. Its characteristics are sprints and predefined roles for people on or around a project. The main roles are Scrum Master, Product Owner and Team. (3)

Scrum Master maintains SCRUM processes and is responsible for removing impediments to the ability of the team to deliver the sprint goal. S/he is not the team leader, but has the task of enforcing rules of SCRUM. Product Owner is best described by being the voice of the customer. S/he is not the customer, rather the interface or buffer between customer and the Team. His/her task is to ensure the Team delivers value to the customer. The Team is a self-organizing, self-led production unit of less than 10 people. It is responsible for delivering the product. (3)

A sprint in SCRUM means one development cycle. Lasting from two weeks to one month, one sprint produces an usable and potentially releasable product. Sprints can be thought of as projects with no more than one month of planned goals, activities and lifetime. All sprints in one product development process have same duration and new sprint is strated immediately after previous sprint ends. A sprint has six activities: one 4 to 8 hour sprint planning meeting at the start of a sprint, 15 minute daily scrum meeting at the start of each work day, the development work, one 2 to 4 hour sprint preview and one 1.5 to 3 hour sprint retrospective at the end of a sprint. (3)

A sprint planning meeting consists of two equal duration parts, which are used to find an answer to these two questions: what will be done in this sprint and how will the work get done? A 15 minute time for daily scrum is used to let each member of the development team explain what has been accomplished since the last daily scrum, what will be done before the next daily scrum and what obstacles are in the way of accomplishing given tasks before the next daily scrum. (3)

A sprint review is for wrapping up the sprint by listing things what has been done and what has not been done, what went well during the sprint and what problems were ran into. Done work is demonstrated to stakeholders and ques-

tions answered about it. The product owner discusses the product backlog and completion dates. The entire group discusses on what to do next as a ground-work for subsequent sprint planning meetings. (3)

3.1.3 Iterative and incremental development

Iterative and incremental methodology is like SCRUM but without sprints or roles (Figure 2). It is a cyclic process developed to address the problems of the waterfall model.

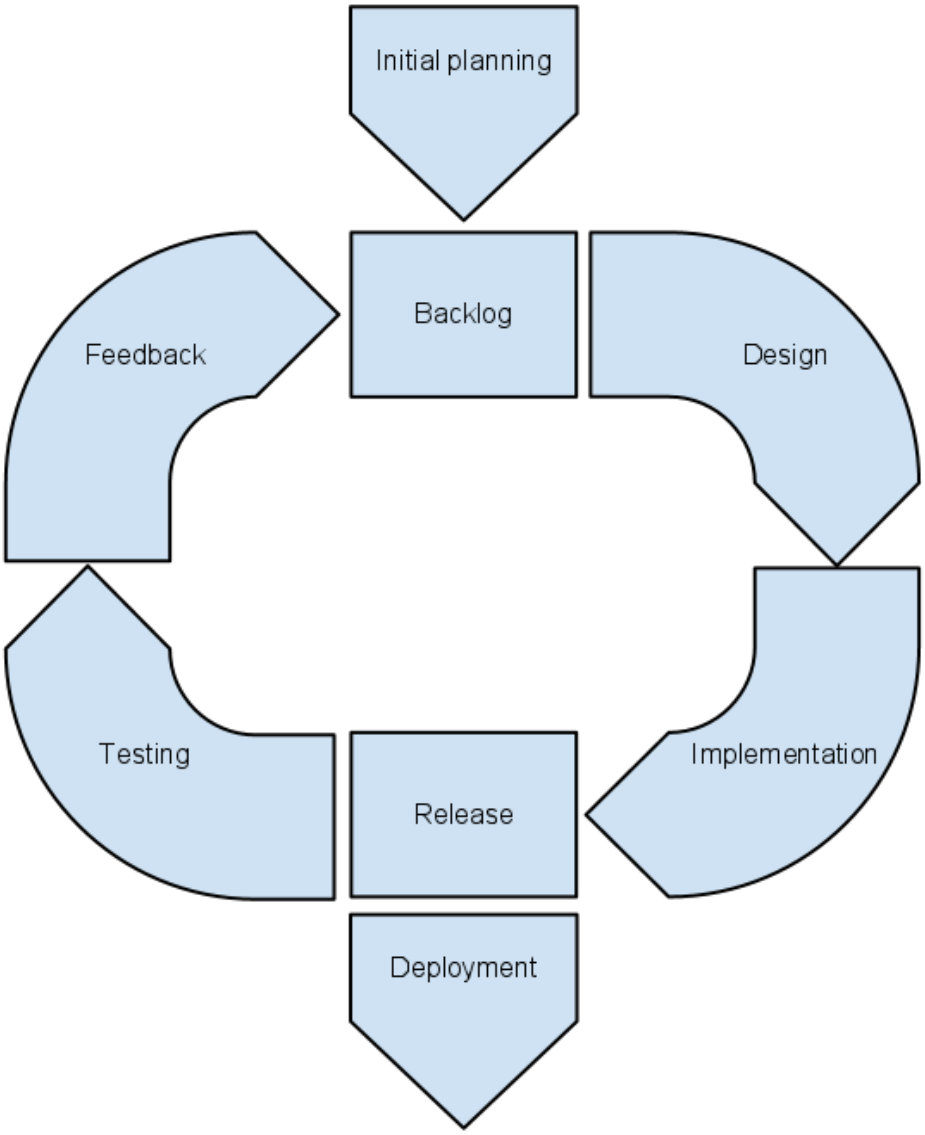

FIGURE 2. Iterative development process

3.1.4 Model-View-Controller, MVC

Model-view-controller is a software architecture used for creating applications with separated aspects and loose coupling between them. The model represents the stored data, data structures and contains functions to read and modify its state. The model's state determines the state of the whole program. The view displays a suitable representation of the state of the model. The controller reads the user input and uses the model's functions to initiate a response and change the state of the model. Model-view-controller comes in different flavors, but generally the control flows from user to controller to model to view and to user again (4, 5, 6, 7) (Figure 3).

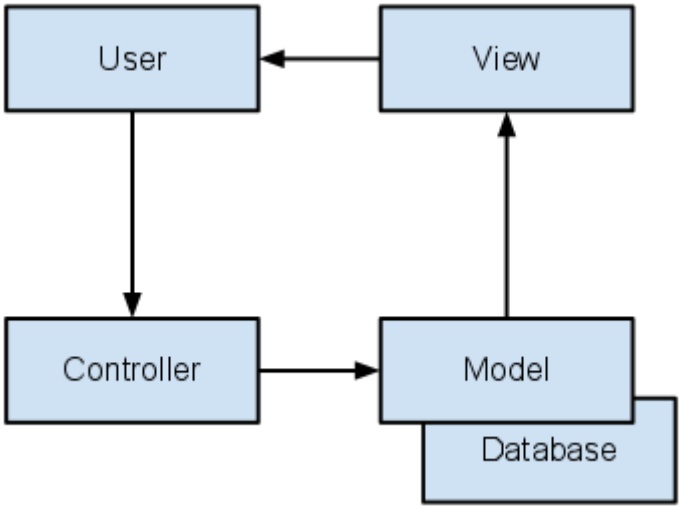

FIGURE 3. Model-view-controller pattern

3.1.5 Model-View-Controller on CodeIgniter

The PHP application development framework used in this project, CodeIgniter, is loosely based on the model-view-controller architecture pattern. The major difference is that the controller acts as an intermediary between all the other resources (Figure 4). The user interacts with the view, using the HTML and JavaScript controls and functions. The view loads and saves information using the controller's functions. The controller uses the model's functions to load and save information to and from the database. (8, 9, 10)

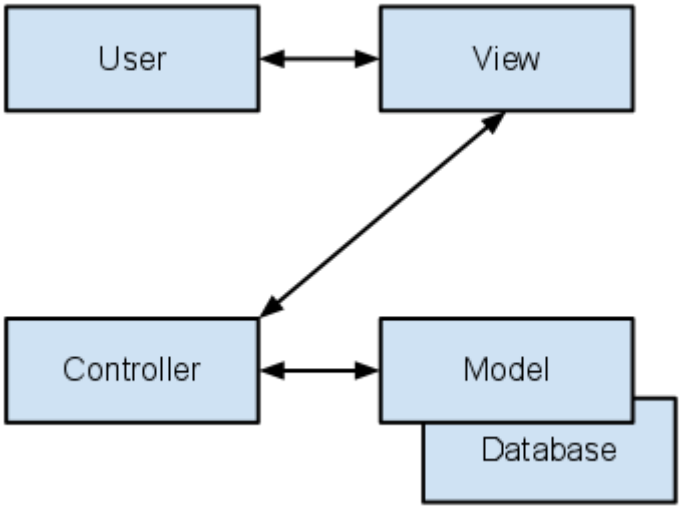

FIGURE 4. Model-view-controller on CodeIgniter

3.2 Practice

After careful consideration and evaluation of available resources and needs to manage the mentioned resources, SCRUM was found too cumbersome for this project. Still, in an attempt to stay within the guidelines set in the beginning of the project and to satisfy a general need for an organized way of working, a decision was made to use only iterative and incremental components from the SCRUM methodology.

Requirements from the client were reviewed and written to the product backlog. From there, the requirements were added to the product, and the first version was released for review and feedback from the client. Using feedback, the product was then modified to meet the requirements better and then released for the second review and feedback round. This cycle was repeated until the project deadline.

4 TECHNOLOGIES AND TOOLS

This chapter describes programs and technologies used in making of the work and how they were used. All programs were most recent versions at the time of working the project.

4.1 HTML4

Hypertext markup language, version 4, is the predominant way to describe web pages for web browsers to render and display. HTML is written in the form of HTML elements consisting of tags, enclosed in angle brackets (like <html>), within the web page content. HTML tags most commonly come in pairs like <h1> and </h1>, although some tags, known as empty elements, are unpaired, for example <img>. The first tag in a pair is the start tag, the second tag is the end tag (they are also called opening tags and closing tags). In between these tags web designers can add text, tags, comments and other types of text-based content. The nested nature of HTML can be seen in its domain object model (DOM) tree (Figure 5). (11, 12, 13)

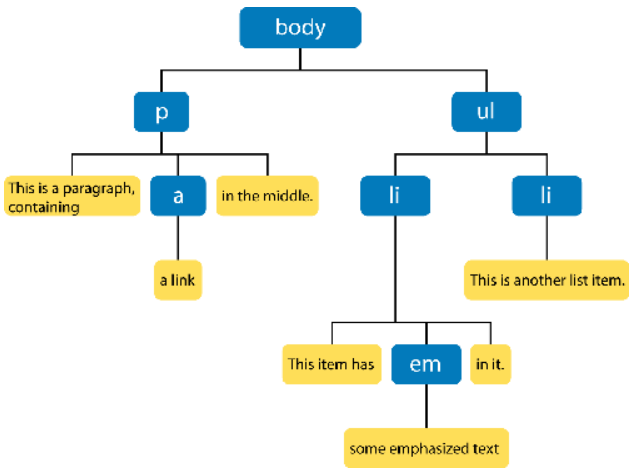

FIGURE 5. An example of a DOM tree (2)

4.2 jQuery JavaScript Framework

jQuery is a cross-browser JavaScript library designed to simplify the client-side scripting of HTML. JavaScript programs made with it are compatible with all major browsers. Without jQuery or a similar JavaScript framework, JavaScript developer would have to detect the user's browsers and their versions, keep track of the existing and new unfixed bugs in their JavaScript implementations, and make browser specific functions to work around the bugs. (14, 15)

4.3 PHP

PHP is a general-purpose server-side scripting language originally designed for web development to produce dynamic web pages. For this purpose, PHP code is embedded into the HTML source document and interpreted by a web server with a PHP processor module, which generates the web page document. Because PHP code is executed server-side, the client will never see the code generating part or all of the HTML content of a web page. PHP can also be used for command line scripting and writing desktop applications using PHP-GTK extension. (4.) (16, 17)

4.4 CodeIgniter PHP Framework

CodeIgniter is an Application Development Framework for building web sites using PHP. It is based on the Model-View-Controller development pattern.(18)

4.5 TCPDF PDF Exporter

TCPDF is a PHP-based library for generating PDF documents. Since PDF is a platform independent document format and an open ISO 32000-1 standard, it is a reliable way to make printable documents and expect identical print results regardless of the hardware or software used. (19.)

Using TCPDF to generate a PDF file is done in five steps:

```
require_once('tcpdf5/tcpdf.php');                // 1. Call the library.
$pdf=new TCPDF();                                // 2. Make a TCPDF object.
$pdf->AddPage($orientation,$papersize);          // 3. Add a new page.
```

```
$pdf->SetFont('Helvetica','B',12);          // 4. Write content.
$pdf->Text(20,10,'Ipsum lorem yadda yadda'); // 4. Keep writing content.
$pdf->Output();                             // 5. Call output function.
```

4.6 XAMPP

XAMPP is a web server application stack. The major components it includes are Apache HTTP server, MySQL database server, Perl and PHP scripting language modules for Apache HTTP. XAMPP Control Panel Application included in the release allows for an easy administration of the various server software packages. (Figure 6.) (20.)

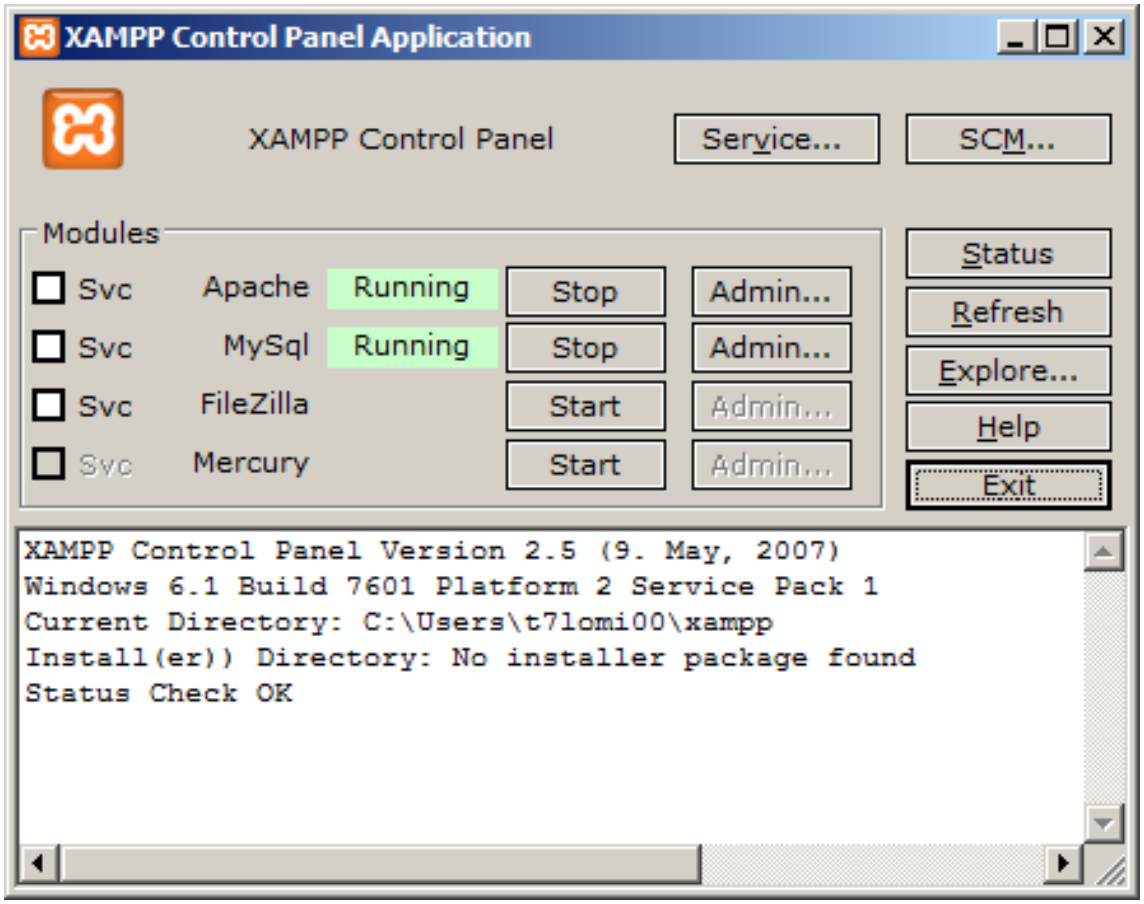

FIGURE 6. XAMPP Control Panel Application

4.7 NetBeans

An integrated development environment (IDE) for developing with Java, JavaScript, PHP, Groovy, C, C++ (Figure 7)(21). It is made with Java and it can be

run on all platforms where compatible Java Virtual Machine is installed. It offers support for third party plug-ins with which its capabilities can be extended when needed.

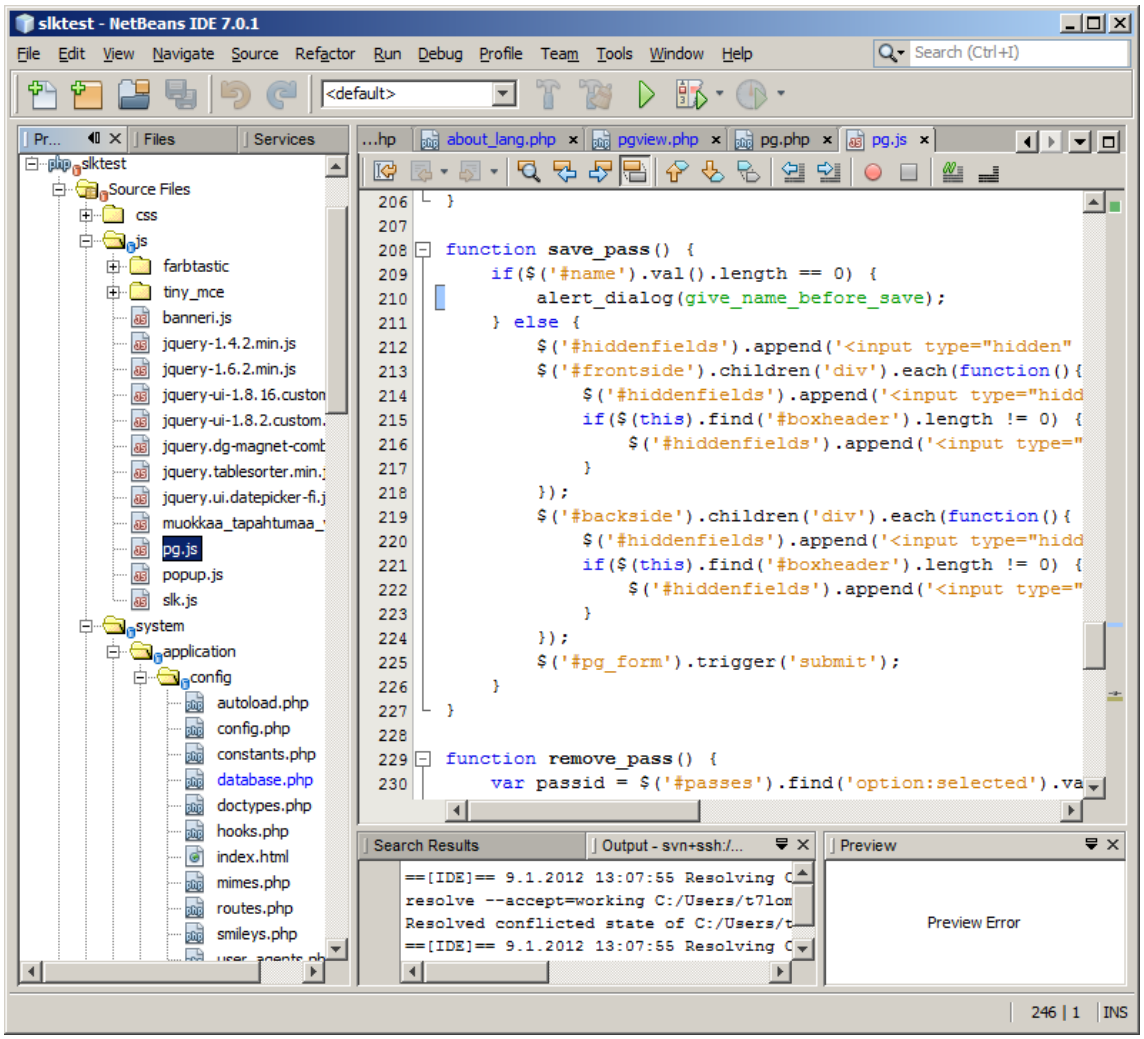

FIGURE 7. NetBeans User Interface

4.8 Google Chrome

Google Chrome is a web browser developed by Google that uses the WebKit layout engine. It is the natural choice for in-development web application testing, because its layout engine has one of the most diverse operating system supports (Figure 8)(22). In addition, it has integrated developer tools for debugging and it is steadily gaining popularity over other major web browsers.

Engine	Windows	Mac OS X	Linux	BSD	Unix	Symbian OS
Gecko	Yes	Yes[note 3]	Yes	Yes	Yes	No
GtkHTML	Yes	Yes	Yes	Yes	Yes	No
iCab	No	Yes	No	No	No	No
KHTML	Yes	Yes	Yes	Yes	Yes	Yes
NetFront	Partial[note 4]	No	Yes	No	No	Terminated[4]
Presto	Yes	Yes	Yes	Yes	Yes	Yes
Prince XML	Yes	Yes	Yes	Yes	Yes	No
XEP[5]	Yes	Yes	Yes	Yes	Yes	No
Robin	Yes	No	No	No	No	No
Tasman	No	Yes	No	No	No	No
Trident	Yes	Dropped (4.0)	No[6]	No	Dropped (5.0)	No
WebKit	Yes	Yes	Yes	Yes	Yes	Yes

FIGURE 8. Operating system support of different layout engines (7)

5 INITIAL PLANNING

Using product backlog as the baseline for the planning, a decision was made to start the project from the user interface and layout design. Those are the foremost elements the users notice, and with a lightweight application, heavy underlying structures they were deemed unnecessary.

The product backlog includes:

- Ability to print on both sides of paper
- Event and person specific id card
- Selected information picked from event sign up database
- Positioning of selected information
- Background image
- Logos
- Can save templates
- Multiple paper sizes

After parsing the backlog, the following user interface elements were needed:

- Selection for 1/2-sided printing
- Event selection
- Information selection from selected event
- Draggable information containers
- Upload and automatic stretching and centering of background image
- Saving and loading of pass templates
- Selection for paper sizes

Additional needed properties for the product are as follows:

- WYSIWYG (what you see is what you get) view of the pass in edit
- Printing via PDF files. Export passes to PDF and use PDF viewer's printing controls for more precise results compared to a print from a web browser
- Selection of font, font style and font color for individual information containers

6 IMPLEMENTATION

This chapter describes the implementation phase of the project. Event pass generator has one controller: pg.php; one model: pgmod.php; and one view: pgview.php. The view has JavaScript functions located in pg.js.

6.1 Controller

The controller in CodeIgniter is the starting point for a new page load process. A web browser accesses the program by pointing to one of the controller files. Project's only controller pg.php includes the functions listed in this chapter.

6.1.1 Function Pg()

Function Pg()'s constructor. Like all controller constructors in Koululiikkuu Suomi project, it calls a check for user privileges. If the current user is not privileged as a logged_in user, the function redirects to the front page. Otherwise it loads the pass generator model, pgmod.

6.1.2 Function index()

Function index() is the default function to run if none were given. First, it fetches lists for the current user's previously saved passes and events. Next, it fetches lists of sign up questions for the found events. After that, it tries to fetch the pass with a given identification number. If a pass is found, it fetches a list of the saved data fields for the pass. Finally it loads and displays the view, pgview, using all the fetched data.

6.1.3 Function form_target()

Function form_target() is the target for any form submits. The view has two different reasons for submitting its form: printing and saving a pass. One of the actions will be selected by a JavaScript function when the send button is pressed. First, a hidden form_action HTML tag with a value of either "printpass" or "savepass" will be inserted in the HTML document and then the submit will be triggered.

When printing, the function first calls for saving the pass, then loads it back to verify the saving. Finally it calls the printing function, _print_pass().

When saving, the function only calls the save_pass function.

6.1.4 Function _collect_data()

Function _collect_data() processes data from the form submits, and it includes possible uploads if any files are given. First, it checks if the pass has previously been saved. If the answer is no, it calls the save_pass function before continuing.

After that, it configures and loads CodeIgniter's upload library, attempts to upload the background image for the front side of the pass and then reinitializes the library and attempts to upload the background image for the the backside os the pass. Next, it gets data from the text fields and parses the CSS values from the data fields and returns them all to the calling function.

6.1.5 Function _print_pass($pass_properties,$fieldsdata)

Pass printing function does not actually print passes with a printer. Instead, it exports them to pg_pdf() function in the PDF exporter helper. PDF viewing programs have sophisticated printing controls, and in this work, it was deemed unnecessary to duplicate such controls in pass generator.

PDF export needs pixels per inch (ppi) resolution of the user's display but, since neither JavaScript nor HTML has any ability to see the user's hardware, the ppi had to be approximated to a value of 94.3/25.4. That value is usual dots per inch value of a monitor divided by millimeters per inch.

Next, the function gets values for double sidedness and paper size and parses needed values from the CSS styles of the data fields. After that, the function fetches all answers for the event which the pass is attached to and reorders them for easier parsing in the PDF export helper, pg_tcpdf_helper. Finally, the function gets uploaded data of the background images, adds it to the properties of the pass and calls the pg_pdf function using all the processed data.

6.1.6 Function remove_pass()

Function remove_pass() simply gets a pass id from the $_POST array and calls the delete_pass function in the model using the id as parameter.

6.2 PDF Exporter Helper

PDF export helper, pg_tcpdf_helper.php, uses the TCPDF library to generate a PDF file. The helper has a pg_pdf function which contains the TCPDF layout and formatting instructions, and the content is passed to it by parameters. The layout code adds crop marks and automatically fits the maximum quantity of the passes a page can hold using the given paper size.

6.3 Model

The model pgmod.php has the following functions:

Function Pgmod(). The constructor includes a simple database query for description of one of the pass generator tables. If the query fails, it is assumed there are no tables for the pass generator. In that case, the constructor calls a function, _create_kipage_tables, to create the needed tables.

Function list_passes($user_id). It makes a database query for all stored passes for a given user. This function is used to populate the saved passes list in the view.

Function list_events($username). It makes a database query for all stored events for a given user name. This function is used to populate the event selection list in the view.

Function list_data_fields($event_id). It makes a database query for all stored registration questions for a given event. This function is used to populate the list of the sign up form questions in the selected event.

Function save_pass($arr,$fieldsarr). It stores a given pass to the database. This function gets two arrays as parameters. The arrays contain lists for text field values, the filenames of the uploaded images and the CSS styles of the data fields.

Function delete_pass($pass_id). It deletes a given pass from the database. The function checks the previously saved background images for the pass and deletes the corresponding files if it finds any.

Function load_pass($user_id,$pass_id). It makes a database query for a specific stored pass for a given user.

Function load_fields($pass_id). It makes a database query for all stored information fields for a given pass.

Function get_last_insert_id(). It fetches the latest auto incremented value from the database and is used after the insert queries. This function contains the MYSQL-specific SQL command. Newer versions of CodeIgniter have a database vendor independent command to accomplish this.

Function get_answers_for_printing($event_id,$pass_id). It makes a database query for all stored participant answers for the registration questions of the given event.

Function _create_kipage_tables($database). It makes tables for the event pass generator and is used only once and automatically after the installation.

6.4 Views

Building an application around the user interface and user experience means building it for a work flow of a typical use case. An attempt was made to design the user interface of Pass Generator for a top-to-bottom work flow. Therefore, the layout is vertically divided in three parts. (Figure 9.)

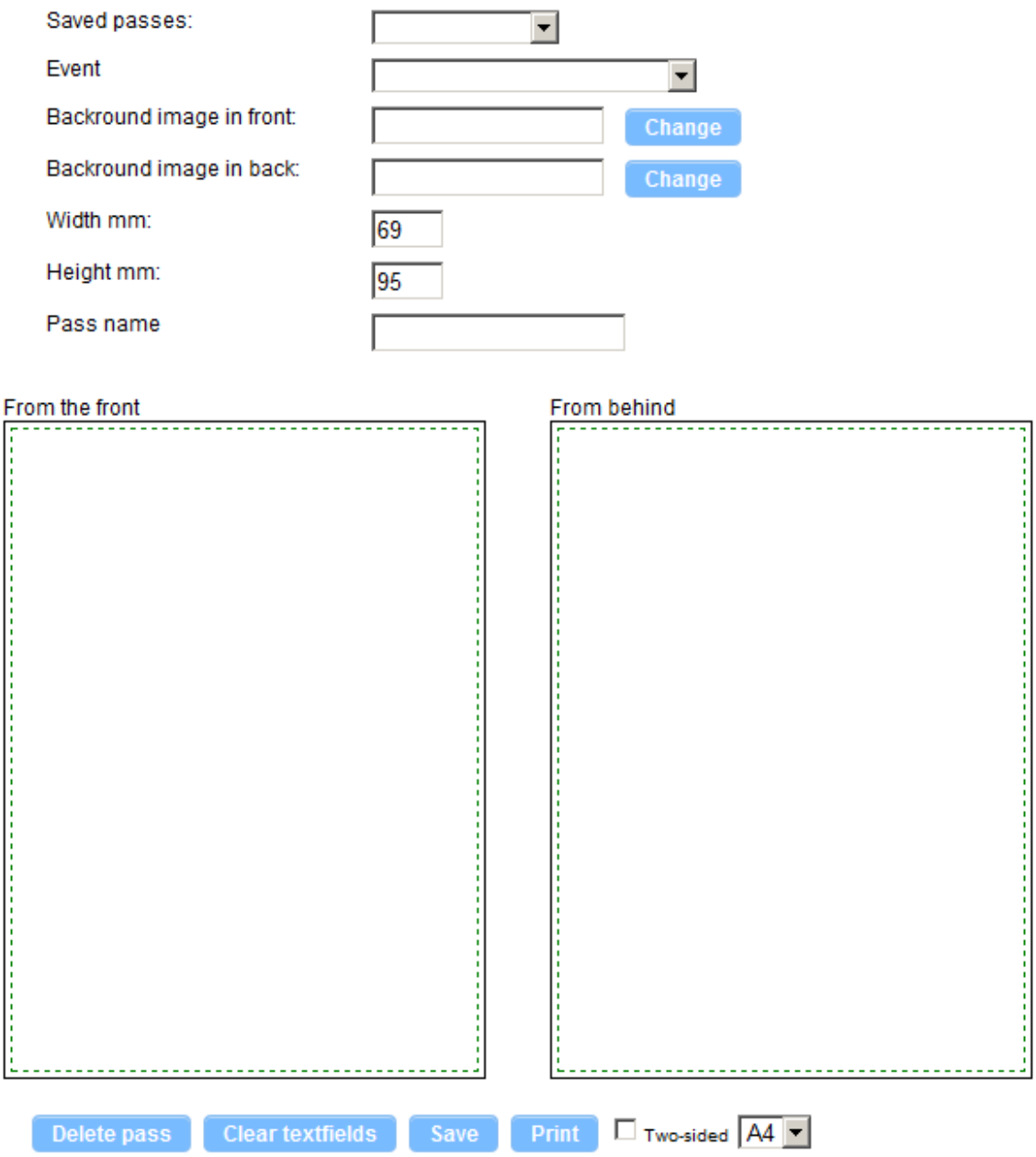

FIGURE 9. View of the pass generator

6.4.1 User interface elements from top to bottom

First element is the dropdown list of the saved event passes. The previously made templates for the event passes are listed in a dropdown list. Choosing one of the saved pass templates loads it and populates the rest of the input fields of the generator with its values. (Figure 10.)

FIGURE 10. Dropdown list for saved events

After that, is the the event selection dropdown list. The events by a currently logged in user are listed in the second dropdown list. Choosing one event from the list brings up a new dropdown list next to it. The new list is populated with questions from the selected event. (Figure 11.)

FIGURE 11. Dropdown list for event selection

Choosing an item from the new list inserts a new text box containing the chosen item to both sides of the pass. The inserted text boxes have controls for text size, header, color, font and style. (Figure 12.)

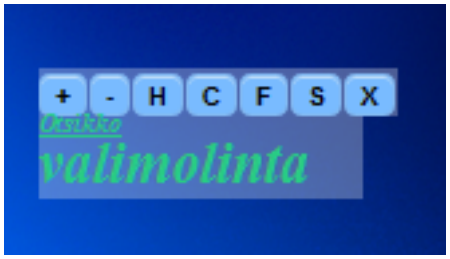

FIGURE 12. The controls for text size, header, color, font and style

Two following input fields are used for uploading and changing the background images for the pass. The images are scaled to fit the dimensions of the pass with 1 millimeter bleed edges.(Figure 13.)

FIGURE 13. Upload controls for background images

Next, two fields are used for setting the pass width and height by millimeters. (Figure 14.)

IMAGE 14. Pass dimension controls

The last text input field is for the name of the pass template. (Figure 15.)

FIGURE 15. Pass name text field

The center area is dedicated for the layout views of a pass. The front and back views are laid side by side if pass width allows.(Figure 16.)

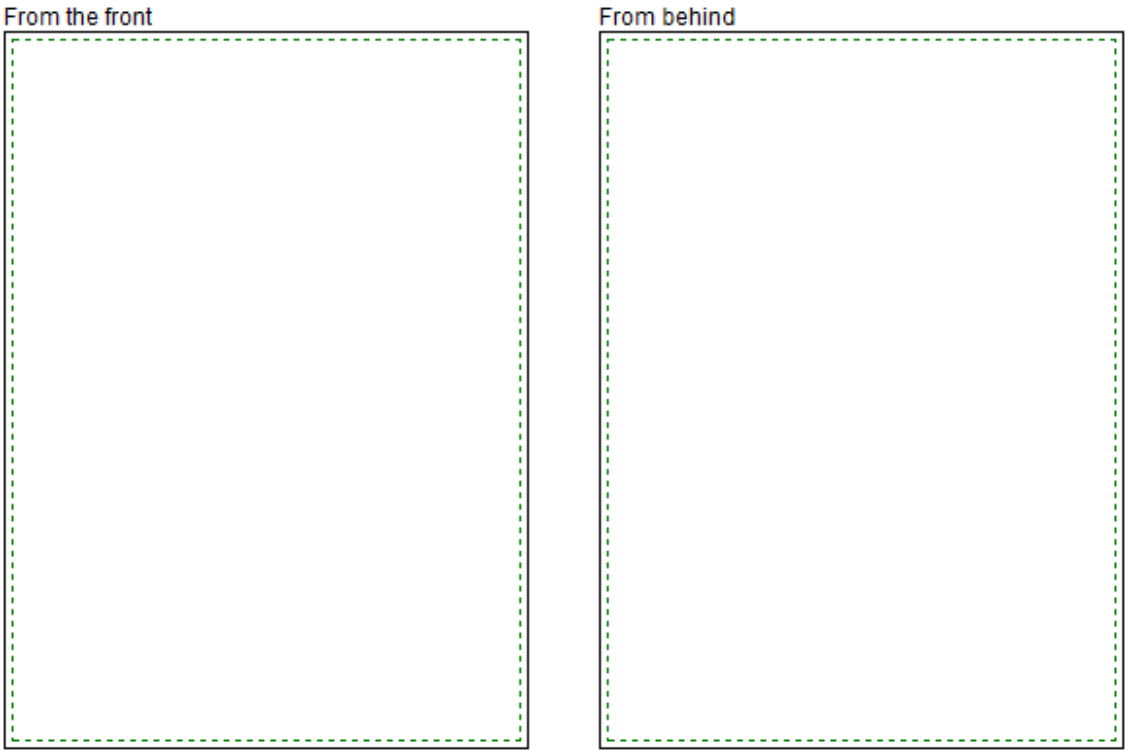

FIGURE 16. Front and back views of a pass

Finally, on the bottom of the screen there are the controls for deleting, clearing text fields, saving and printing the selected pass. Next to the print button there

are the checkbox for two-sided printing and a dropdown list for the paper size. (Figure 17.)

FIGURE 17. Buttons for delete, clear, save and print

6.5 JavaScript

The JavaScript functions used in the work are as follows:

document_ready(). This function runs after every page load. It has the initialization of Farbtastic color picker, workarounds for Internet Explorer and an event binding for the event selection control.

The event bind opens a dialog telling that the changing event clears the already filled text fields and asks if the user really wants to continue. It inserts the Yes and No buttons to the dialog for an easier answer.

change_pass_x_size(). This function alters the layout views to match the given new width for the pass.

change_pass_y_size(). This function alters the layout views to match the given new height for the pass.

change_bg(front). This function changes the background images. It takes a Boolean parameter to decide if the image to change is in the front or back side of the pass.

load_fields_list(event_id). This function populates the questions list for a given event. It first removes any text boxes already set in the layout views. Then it

hides any question selection dropdown list, and if any event is selected, it displays the selection dropdown of the corresponding questions.

The HTML document has lists for questions from all of the user's events ready and hidden. When the load_fields_list() function is run, it only hides one list and displays another if another event is selected. In this way, an additional AJAX is avoided call every time a new list is needed.

create_field_box(id,content,style,side,header) This function creates one data field and its edit control buttons on a layout view. (Figure 18.) The function first saves the needed HTML code for the buttons in to a variable, btnpanel.

The buttons are a plus sign (+) for enlarging the font, a minus sign (-) for shrinking the font, H for toggling the header on or off, C for showing the color picker for selecting the text color, F for switching to another font, S for switching to another text style and finally X for deleting the data field altogether.

After appending the data field box in one of the layout views, the function resets the dropdown list of the questions to display an empty row again and calls the add_hover_events() function.

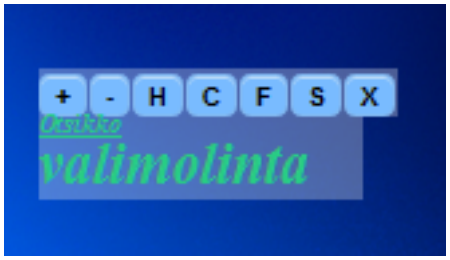

FIGURE 18. Data field and its edit control buttons

add_hover_events(). This function adds two mouseover events for the edit controls. The create_field_box function calls this after creating a new data field box. The first hover event displays the edit buttons for the data field when the mouse pointer is moved over the text. The second hover event is triggered when the

mouse cursor leaves on of the layout view areas. The event hides all visible edit buttons.

bigger_font(ptr). This function enlarges the font of a data field. This function adds two pixels to the font size of the selected data field text.

smaller_font(ptr). This function shrinks the font of a data field. This function removes two pixels from the font size of the selected data field text.

toggle_header(ptr). This function displays or hides the header of a data field. This function inserts or removes a new DIV with the necessary content to show a header for the selected data field.

font_color(ptr). This function changes the font color of a data field. It uses Farbtastic jQuery plug-in to do this. First, it links the target element with Farbtastic using the linkTo-method. Then, it sets Farbtastic to show the target element's current color, and finally it opens the Farbtastic dialog.

change_font(ptr). This function changes the font of a data field. The function toggles between three font families: serif, sans-serif and monospace. It looks up the currently used font and returns the next one in its list. If no current font is detected, it defaults to the serif.

change_font_style(ptr). This function changes a data field's font style. Similar to the change_font function, this looks up the currently used font style and returns the next one in its list. However, since the font weight and font style are two different parameters in the CSS code, the function needs to toggle through the combinations of normal style, italic style and bold weight, normal weight. In all, that makes four different cases. The function defaults to normal, that is, to the normal style and normal weight.

remove_box(ptr). This function removes a data field.

save_pass(). This function checks the input values before attempting to save a pass. Currently only pass name is a mandatory input. The user sees a save button but that button actually runs this function instead of outright submitting the form.

Because the HTML submit-method only transfers values from the form input fields and because the needed data fields are not input fields, it is necessary to convert the style information of the data fields to the hidden input fields before submitting the form.

After inserting the relevant style information to the inputs for submitting, the function then triggers the form submit.

remove_pass(). This function displays a confirmation dialog to delete a pass. The dialog has Yes and No buttons and a question asking if the user really wants to delete a pass.

alert_dialog(str). This function displays an error dialog. The content of the dialog is passed to it with a string parameter. The dialog has only one Ok-button.

print_pass(). This function checks the input values before attempting to print a pass. Currently, only pass name is a mandatory input. The function is almost similar to the save_pass function. The only differences are an additional hidden input for the screen resolution value and the form action, which is a printpass instead of a savepass.

7 CONCLUSION AND DISCUSSION

The completed event pass generator gives a what-you-see-is-what-you-get editing capability of the passes. It is simple yet flexible enough to allow for making of professional looking passes and ID cards. The event organizers save days of work time while still retaining more flexibility than what they would get from the offers of the printing houses.

The project gave a better understanding to the user interface programming with JavaScript. In addition, it was yet another attempt with the SCRUM project management, giving a better view of its strengths and weaknesses. While SCRUM as a whole still took the backseat, its iterative and incremental part proved useful for this one man project.

At the personal level, I feel I learned more programming during this project and the preceding practice sections than during all other courses. I probably could have got more experience in the half of the time if I would have searched for a suitable job instead of going to school four years ago.

Old habits die slowly and that is twice as much true for the academic world. It does not matter if we are students of communications and information technology, we are still required to submit our theses using the old paradigms. We have to submit paper prints for review and use A4 pagination instead of wiki style non-paginated format despite the fact that we will publish these to an electronic archive. The archives are clumsy collections of PDF files instead of open wikis with cross linked content searchable with Google. We have to use cumbersome word processors instead of lightweight HTML content editors. The mandated format for theses is outdated when compared side-to-side with Wikipedia.

8 LIST OF REFERENCES

1. Software development process. 2012. Available: http://en.wikipedia.org/wiki/Software_development_process. Accessed 27.2.2012.

2. Agile software development. 2011. Available: http://en.wikipedia.org/wiki/Agile_software_development#Agile_Manifesto. Accessed 21.11.2011.

3. The Scrum Guide. 2011. Available: http://www.scrum.org/storage/scrumguides/Scrum_Guide.pdf. Accessed 24.1.2012.

4. MVC XEROX PARC 1978-79. 2012. Available: http://heim.ifi.uio.no/~trygver/themes/mvc/mvc-index.html. Accessed 24.1.2012.

5. Applications Programming in Smalltalk-80(TM):

6. How to use Model-View-Controller (MVC). 1997. Available: http://st-www.cs.illinois.edu/users/smarch/st-docs/mvc.html. Accessed 24.1.2012.

7. Model View Controller. 2011. Available: http://aspnet4.com/asp-net-4-thoery/model-view-controller/. Accessed 24.1.2012.

8. Model-View-Controller. 2011. Available: http://codeigniter.com/user_guide/overview/mvc.html. Accessed 24.1.2012.

9. MVC. 2012. Available: http://codeigniter.com/wiki/MVC. Accessed 24.1.2012.

10. How I use CodeIgniter's MVC. 2012. Available http://www.jimohalloran.com/2007/09/06/how-i-use-codeigniters-mvc/. Accessed 24.1.2012.

11. DHTML Utopia: Modern Web Design Using JavaScript & DOM. 2005. Available: http://www.sitepoint.com/dhtml-utopia-modern-web-design/. Accessed 9.1.2012.

12. HTML. 2011. Available: http://en.wikipedia.org/wiki/HTML4. Accessed 21.11.2011

13. Introduction to HTML 4. 1999. Available http://www.w3.org/TR/html401/intro/intro.html. Accessed 24.1.2012.

14. jQuery. 2011. Available: http://en.wikipedia.org/wiki/Jquery. Accessed 21.11.2011

15. Tutorials: How jQuery Works. 2010. Available: http://docs.jquery.com/How_jQuery_Works. Accessed 21.11.2011

16. PHP. 2011. Available: http://en.wikipedia.org/wiki/PHP. Accessed 17.11.2011.

17. PHP Manual. 2012. Available: http://www.php.net/manual/en/. Accessed 24.1.2012.

18. CodeIgniter. 2011. Available: http://en.wikipedia.org/wiki/Codeigniter#CodeIgniter. Accessed 21.11.2011.

19. TCPDF - PHP class for PDF. 2012. Available: http://sourceforge.net/projects/tcpdf/. Accessed 27.2.2012.

20. XAMPP. 2012. Available: http://www.apachefriends.org/en/xampp.html. Accessed 27.2.2012.

21. NetBeans IDE 7.1 Features. 2012. Available: http://netbeans.org/features/index.html. Accessed 27.2.2012.

22. Comparison of web browser engines. 2011. Available: http://en.wikipedia.org/wiki/Comparison_of_web_browser_engines#Operating_system_support. Accessed 24.11.2011.