Basic support for tables in Scribus

Elvis Stansvik
<elvstone@gmail.com>

March 29, 2011

Contents

Abstract

The Scribus software currently lacks proper support for laying out content in a tabular arrangement, a feature commonly known as tables. In the following document I propose to implement a first basic support for tables in Scribus. The proposal comes in three parts. First, a problem description is given, where I provide a rationale for the project. Second, an implementation plan, including project deliverables and a contingency plan. Third, a tentative timeline for the project, and last some biographical information and other practicalities.

1 Problem Description

Tables provide a compact way of representing tabular information in a page layout, and is a common design element in books, magazines and advertisements. Scribus currently has a shortcut for creating an automatically grouped table-like array of text frames. These are not proper tables. To stay competitive, Scribus needs better support for tables. Support that enables the user to create and edit tables as distinct items with their own set of properties as commonly found in other table implementations.

2 Implementation Plan

Having been through Summer of Code once before, I think I have a good perspective on what's important going into a project of this nature. One thing that cannot be stressed enough is the importance of having something tangible and working by the end of the project period. Not only will this give closure and a sense of accomplishment to me as a student, it also gives the Scribus core developers a good outlook when it comes to taking the code in and building further on it. To mitigate the risk of ending up with an unfinished project, a contingency plan is included. The contingency plan lists features that can be dropped from the project, should time start to run out.

2.1 Basic Idea

The implementation will be done by introducing a new page item, `PageItem_Table`. This item will be responsible for layout and drawing of a table. The layout and drawing of table cell content will be delegated to standard text frames, which might have to be given new features/modifications to better serve this purpose. As content in the table cells change, the table will be notified of this and perform a relayout of the table, starting from the row in which the change occured.

2.2 Project Deliverables

These are features that I believe are realistic to implement during the course of the project (but see further down for a contingency plan). For more detailed chronological information, see the timeline.

- Tables
  - Insert, select, resize and remove.
  - Background color.
  - Border color.
  - Border style.
  - Default cell style.
  - Breaking across page boundaries.
  - Import from CSV and ODT.
- Table Rows
  - Insert, select, resize and remove.
  - Background color.
  - Default cell style.
- Table Columns
  - Insert, select, resize and remove.
  - Background color.
  - Default cell style.
- Table Cells
  - Select and resize.
  - Background color.
  - Border color.
  - Border style.
  - Padding and margin.
  - Row and column spanning.

For all features, load/save roundtrip to Scribus .sla format will be supported.

2.3 Contingency Plan

If unforeseen difficulties are encountered during the project, one or more of the following features may be dropped. Hopefully to be implemented later. Note that should any of these features be dropped, an effort will still be made to keep the feature in mind, to avoid painting ourselves into any corners.

- **Breaking across page boundaries**
  This is a feature which I did not finish completely during my 2009 project for KWord, and one which can be quite hairy to implement. I think I have a good general idea of how to go about it, but should it turn out to take too long, it may be dropped.
- **Row and column spanning**
  This I really think I should be able to pull off. But should time run short, it may be dropped.
- **Import from CSV and ODT**
  While hopefully not too challanging, technically, this feature may be time consuming and is something which can be added later. Hence it may be dropped.

3 Timeline

Below follows a rough tentative timeline for the project. The timeline is of course subject to change, should some parts of the project turn out to be more/less time consuming than anticipated.

I have my last exam on May 28, and my first day of school after the summer is Aug 28. This means my project period will be cut five days short in the beginning. It was the same thing during my project in 2009, and I believe it's quite common for European students taking part in Summer of Code. It just means the timeline has to take this into account.

In the time leading up to the project period, I'll try to acquaint myself with the Scribus code I'll be touching and experiment with data structures and layout algorithms for the tables I'll be implementing.

- **Week 1** May 29—June 5
  - Data structures for tables.
- **Week 2** June 6—June 12
  - Basic table layout with fixed column widths and mock content in cells.
  - Insertion/removal of rows/columns.
  - Basic drawing of table.
- **Week 3** June 13—June 19

  - More advanced layout.
  - Variable column widths.
- **Week 4** June 20—June 26
  - Cell formatting properties such as border and padding.
  - Cell content layout; use Scribus' text frames to populate cells.
- **Week 5** June 27—July 3
  - More integration of text frames. Fixing of issues/road bumps.
  - UI: Row/column selection/insertion/removal/resizing.
- **Week 6** July 4—July 10
  - Wrapping up for mid-term evaluation.
  - Fix bugs/finish up what I have so far.
- **Week 7** July 11—July 17
  (Mid-term evaluation)
  - Row and column spanning.
- **Week 8** July 18—July 24
  - Table/row/column/cell styles.
- **Week 9** July 25—July 31
  - UI: Direct formatting.
  - UI: Integration of styles in Style Manager.
- **Week 10** August 1—August 7
  - Table breaking.
- **Week 11** August 8—August 14
  - More table breaking.
  - CSV and/or ODT import.
- **Week 12** August 15—August 21
  (Suggested pencils down)
  - Wrapping up for final evaluation.
  - Fix bugs/finish up what I have so far.
- **Deadline** August 22
  (Firm pencils down)

4 Post-Summer of Code Commitment

A big part of what makes a successful project is the student's level of involvement after the project period is over. Although the time I will have available for FOSS hacking will go down drastically after the summer, I'll of course stick around to answer questions, fix bugs and help out the best I can. I've stayed in contact with the KWord developers ever since my project in 2009, and I won't treat you Scribus folks differently. In short; I won't disappear on you.

5 Student Biography

Living in Stockholm with my girlfriend Hanna, I'm a 27 year old freshman at the Computer Science program at KTH Royal Institute of Technology in Stockholm. I've dabbled with Free Software for about 12 years now. In recent years KDE has been my main focus and in 2009 I implemented basic support for tables in the KWord word processor as part of Google Summer of Code. Previous employments include working at my dads small publishing company Nya Doxa AB and running my own book shop as well as a book café. I should note that Scribus has been a great help at both these positions. Apart from my studies I enjoy hiking our small Swedish mountains when I have the opportunity, and hanging out discussing with my friends over a beer. I have no other engagements this summer apart from Summer of Code.

6 Contact Information

Name: Elvis Stansvik
E-mail: elvstone@gmail.com
IRC: estan @ irc.freenode.net
Phone: +46763091545