



Notes for the Workshop on Combination Testing

Cem Kaner

January 2006

Copyright (c) Cem Kaner 2006

This work is licensed under the Creative Commons Attribution-ShareAlike License. To view a copy of this license, visit <http://creativecommons.org/licenses/by-sa/2.0/> or send a letter to Creative Commons, 559 Nathan Abbott Way, Stanford, California 94305, USA.

These notes are partially based on research that was supported by NSF Grant EIA-0113539 ITR/SY+PE: "Improving the Education of Software Testers." Any opinions, findings and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

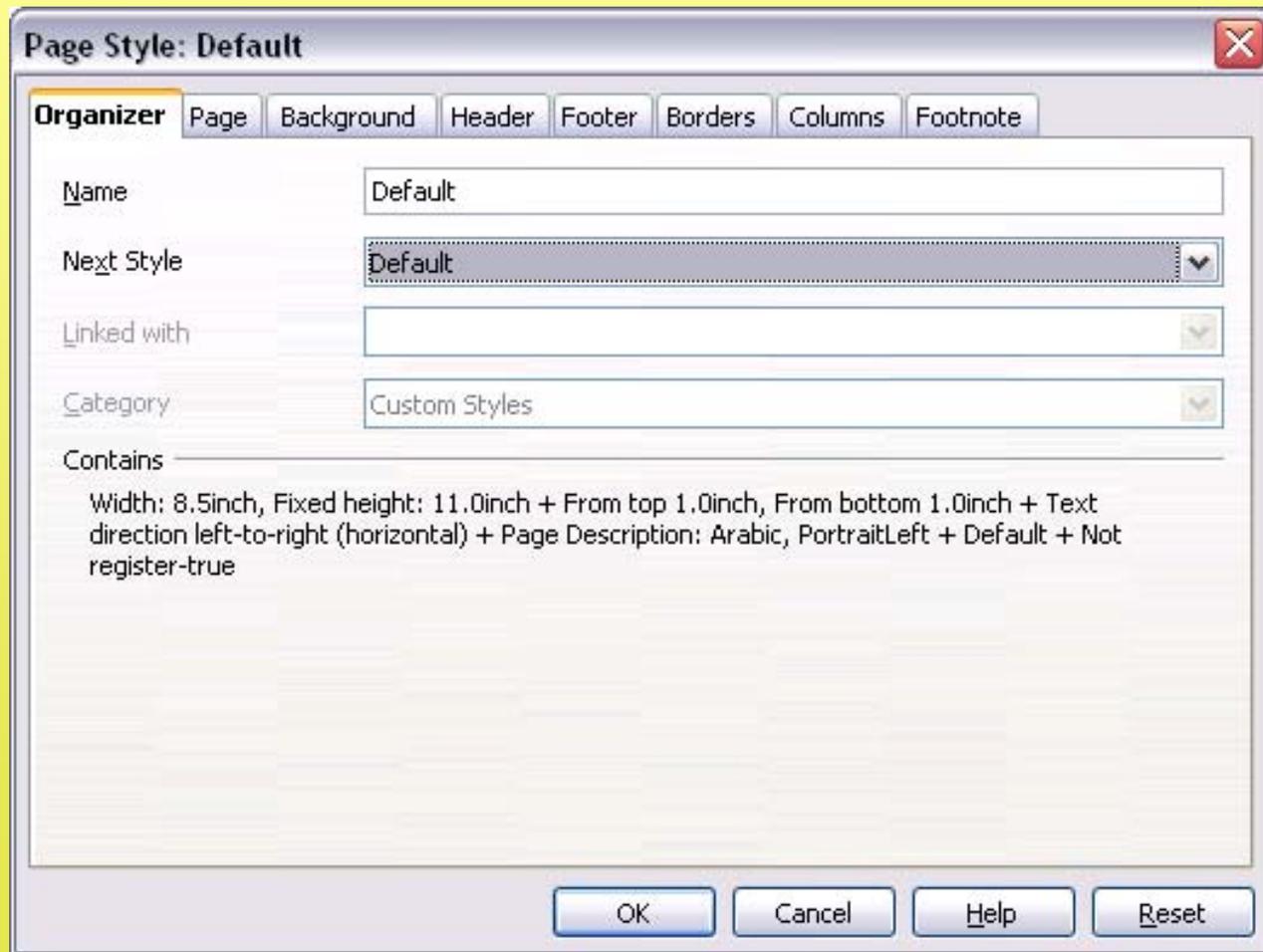


Combination Chart

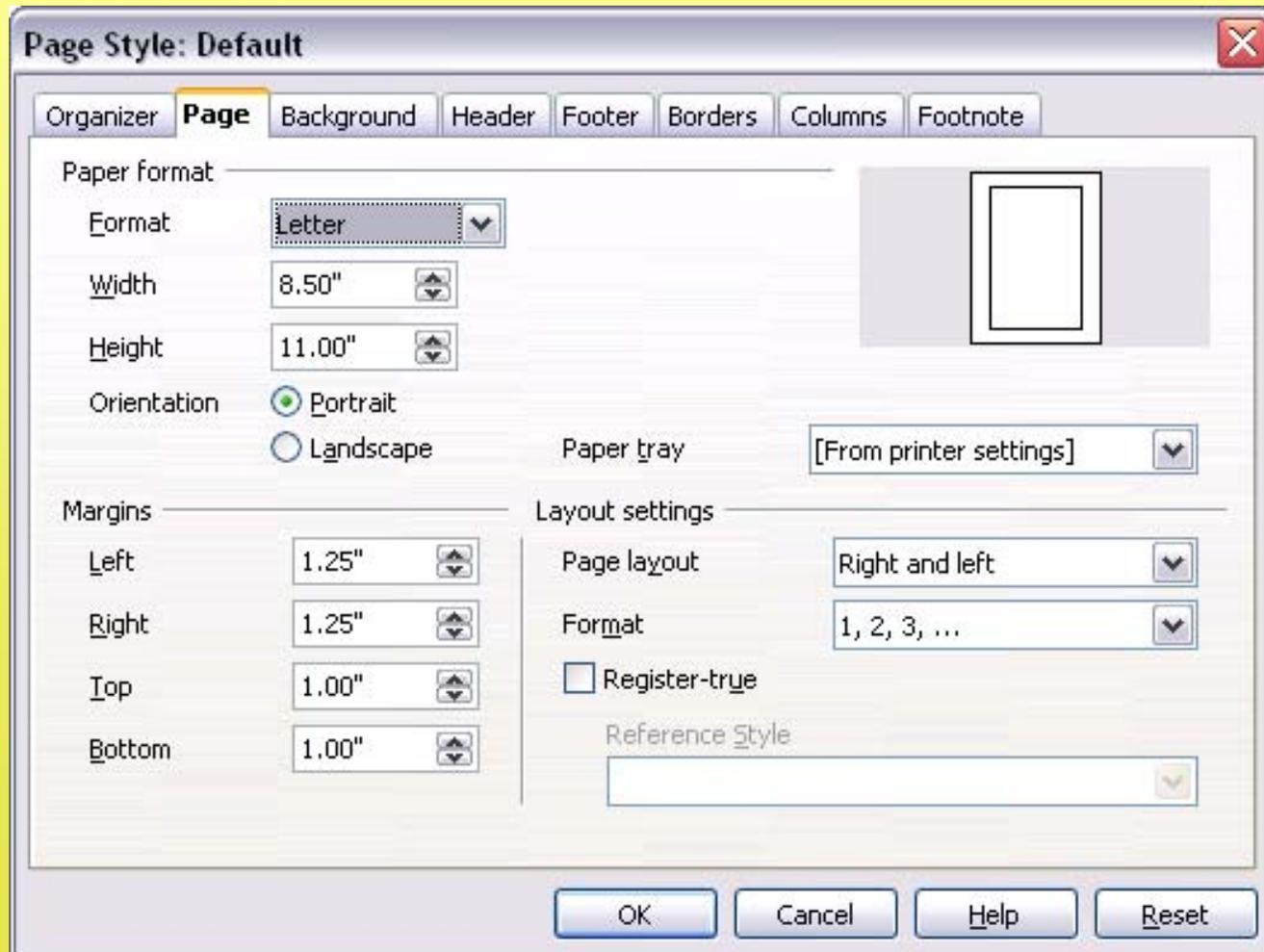
	<i>Var 1</i>	<i>Var 2</i>	<i>Var 3</i>	<i>Var 4</i>	<i>Var 5</i>
<i>Test 1</i>	<i>Value 11</i>	<i>Value 12</i>	<i>Value 13</i>	<i>Value 14</i>	<i>Value 15</i>
<i>Test 2</i>	<i>Value 21</i>	<i>Value 22</i>	<i>Value 23</i>	<i>Value 24</i>	<i>Value 25</i>
<i>Test 3</i>	<i>Value 31</i>	<i>Value 32</i>	<i>Value 33</i>	<i>Value 34</i>	<i>Value 35</i>
<i>Test 4</i>	<i>Value 41</i>	<i>Value 42</i>	<i>Value 43</i>	<i>Value 44</i>	<i>Value 45</i>
<i>Test 5</i>	<i>Value 51</i>	<i>Value 52</i>	<i>Value 53</i>	<i>Value 54</i>	<i>Value 55</i>
<i>Test 6</i>	<i>Value 61</i>	<i>Value 62</i>	<i>Value 63</i>	<i>Value 64</i>	<i>Value 65</i>

In a combination test, we test several variables together. Each test explicitly sets values for each of the variables under test. It's common to represent the pool of tests in a table, like this one.

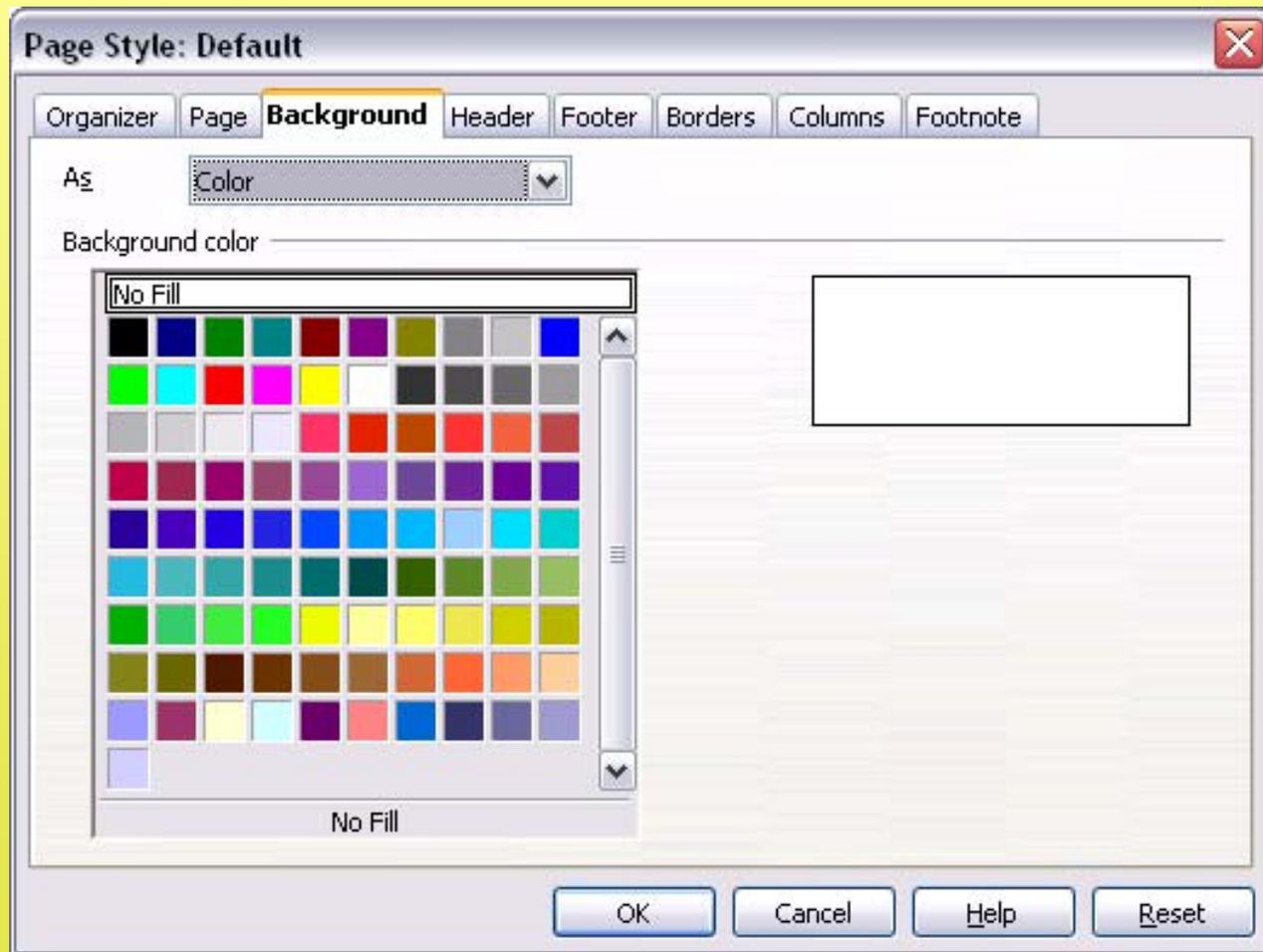
Combination tables (and the all-pairs technique) are nice, but watch how hard this is to apply it to Open Office Writer page style dialog.



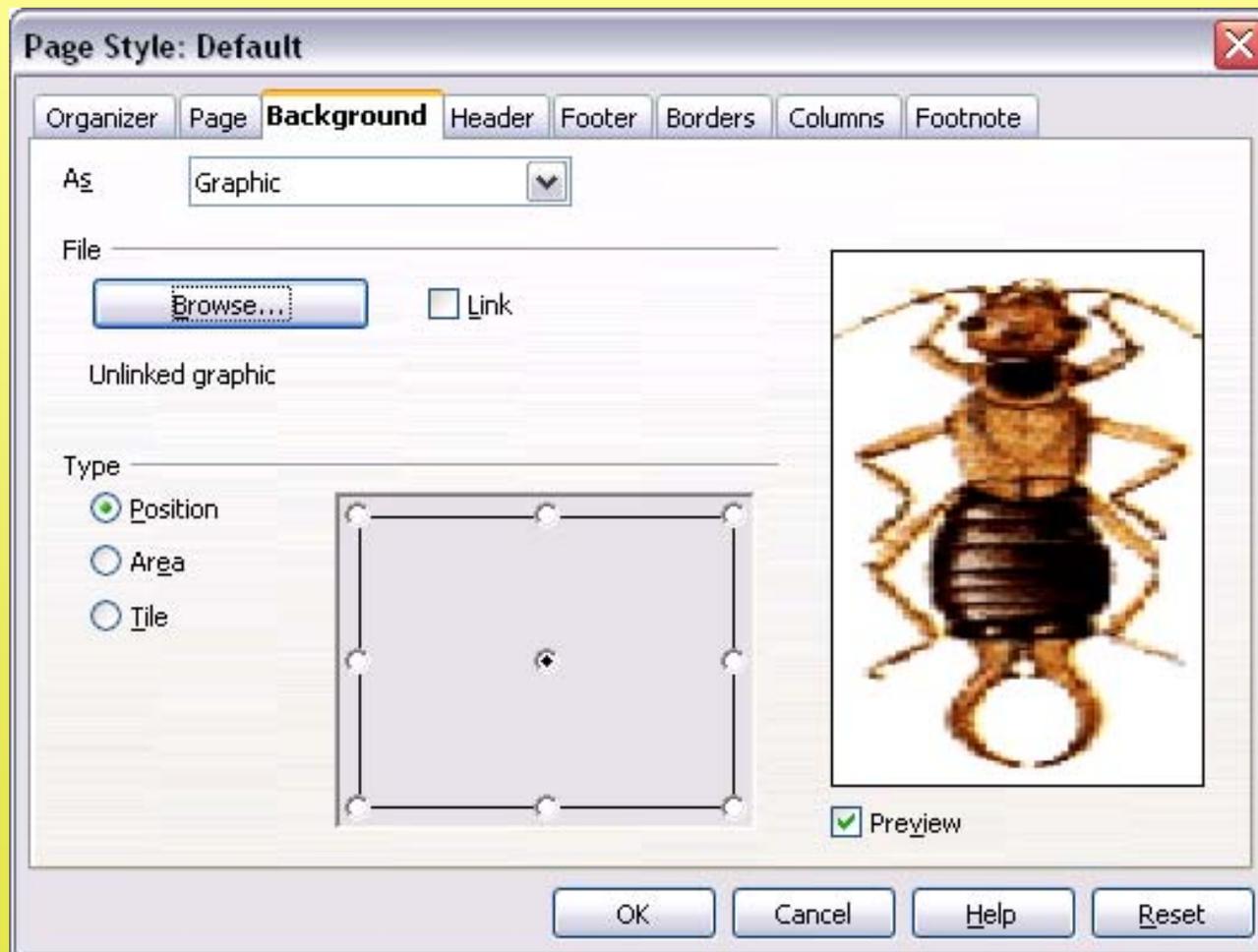
These tabs are interesting as a group because they all interact.



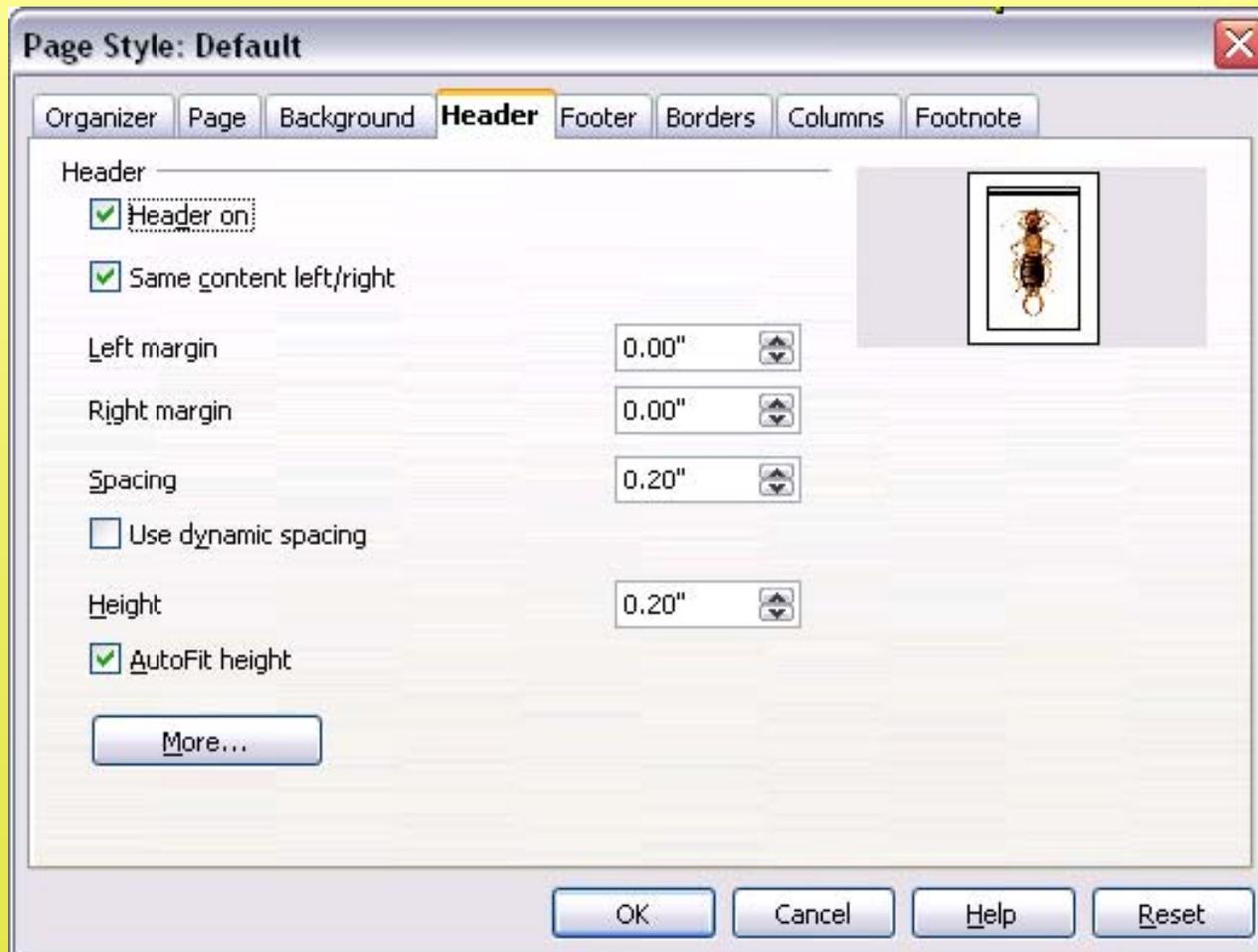
Just print a page. Its layout is jointly determined by all of these



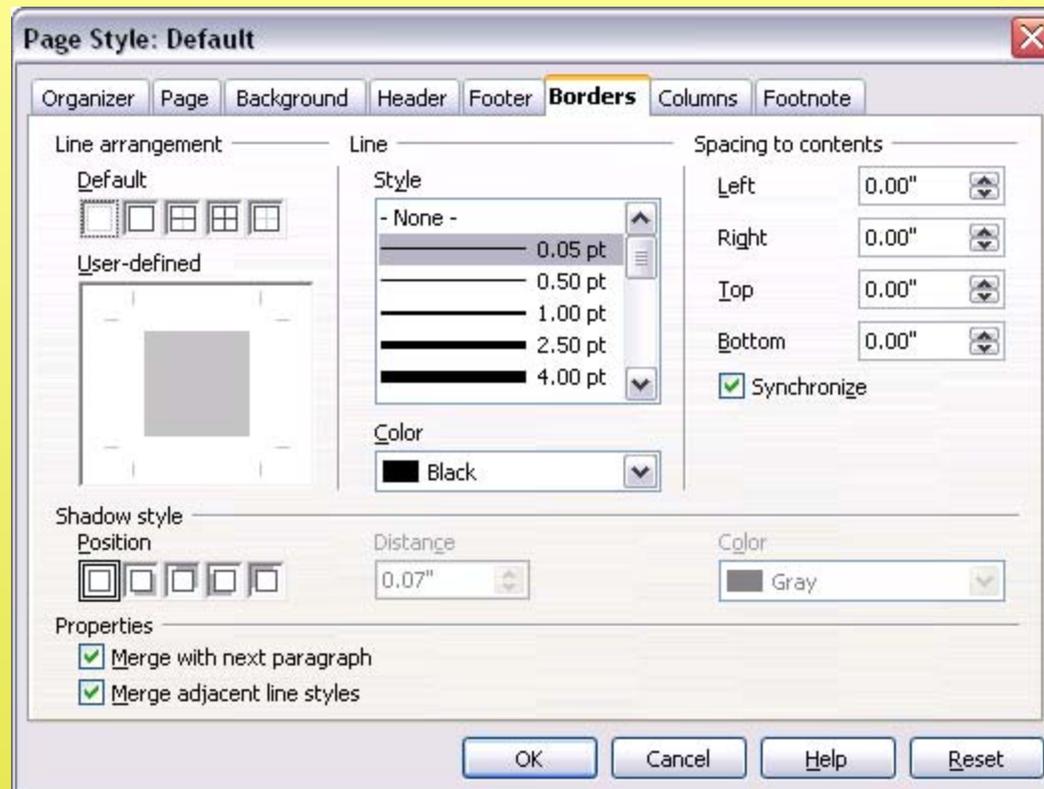
So how do we test these in combination?



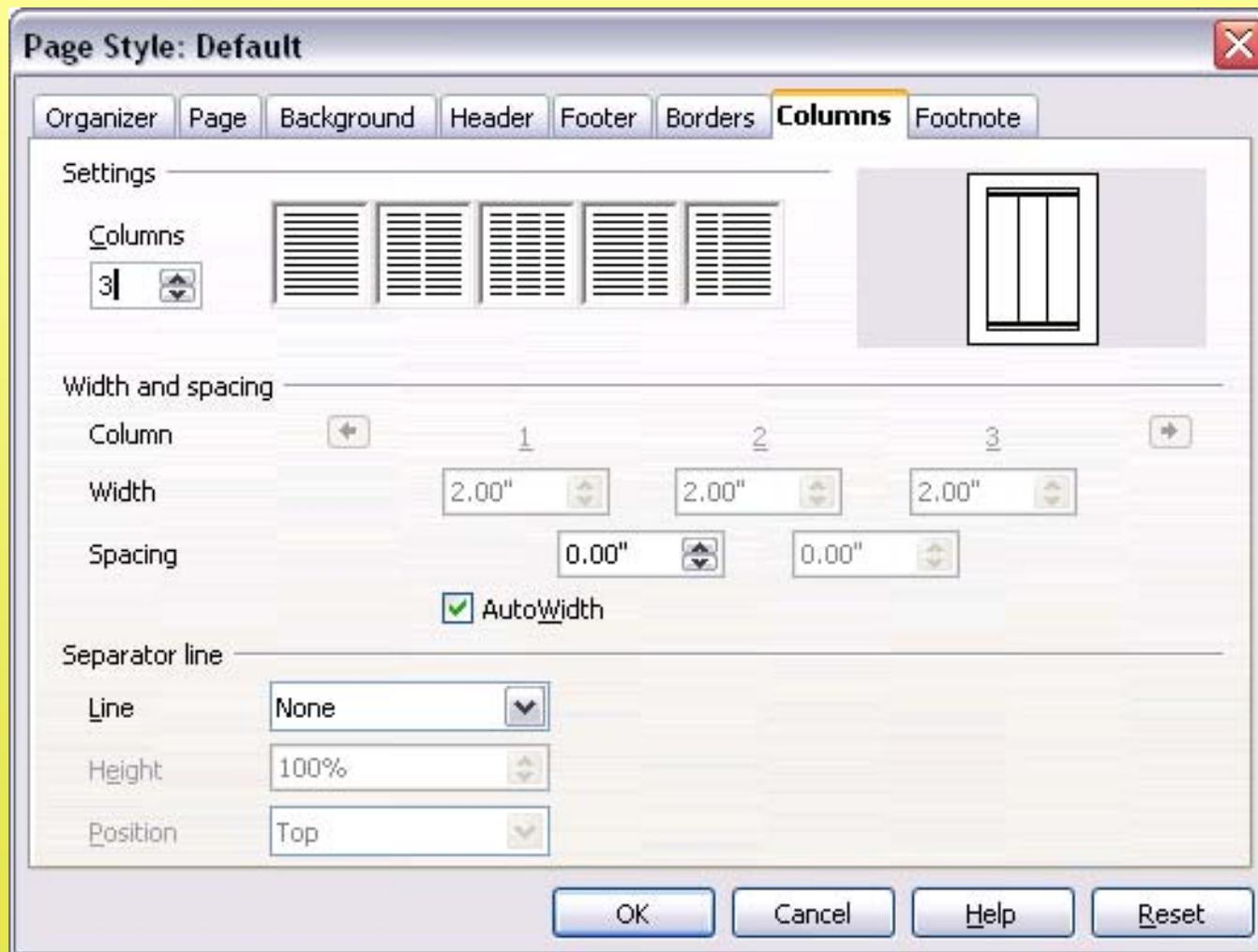
Can you list all the relevant variables?



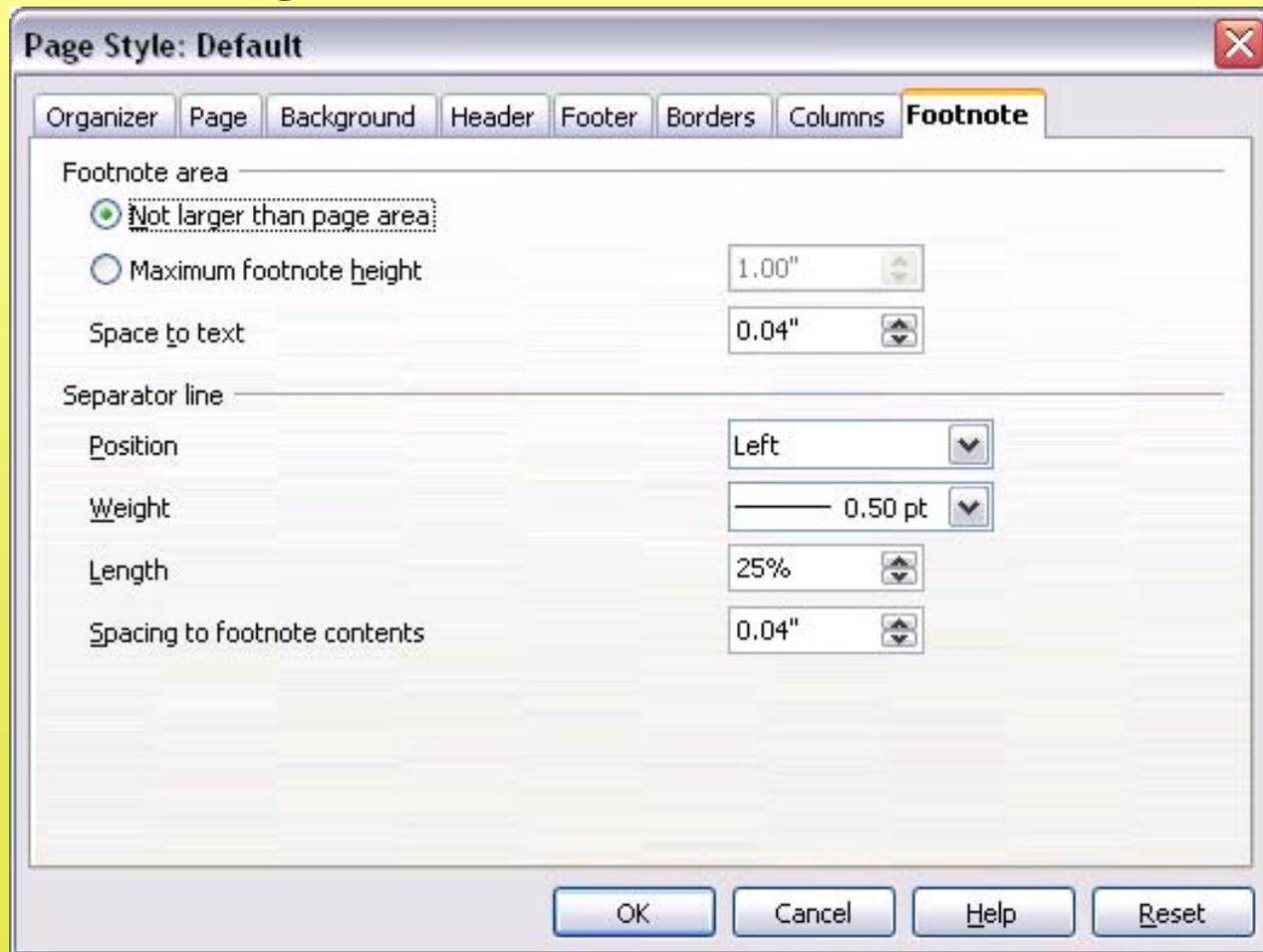
How many variables are on this page?



Number of variables on this page depends on how many columns you choose.



At last, we're through this one (1) dialog.
You can see why people would give up and
do all singles or random combination.



Strategy:

Mental Models of Combinations

Rather than force-fitting a problem to a mental model (like a combination table, useful though it often is), let's should allow different representations of combinations.

What does a combination look like?

- *Combination chart*
- *Sequence diagrams*
- *Mental map (concept map, for example) with dynamically growing pool of nodes and cross-linking of topics*
- *Pool of user stories (XP short form) / use cases (RUP scenario) / user scenarios (Carroll full story with motivation)*
- *What other useful possibilities?*

Note: If you can't map the full set of possible tests easily to a combination chart, you can't easily use combinatorial methods like all pairs to optimize the chart.

Strategy: Information Objectives

Underlying all of these possibilities is the theory of value that drives your testing in the first place. We drive testing differently depending on the type of information that we are trying to discover. So ... ?

- Find bugs*
- Regulatory compliance demonstration*
- Coverage assessment / guarantee*
- Reduce cost of support calls / warranty claims*
- Cover the market space*

Questions of Strategy

- **Mechanical (or procedural).** *Use a routine procedure to determine the tests.*
 - *In an important sense, I think that random testing, combinatorial testing, state-model testing (etc.) are equivalent.*
- **Risk-based.** *Combine test values (the values of each variable) based on perceived risks associated with noteworthy combinations*
- **Scenario-based.** *Combine test values on the basis of interesting stories created for the combinations.*
- **Exploring for Discovery.** *Use exploration of combinations as an active-reading (active-analysis) vector for developing tests and the approach to test.*
- **Assessment of a Theoretical Model.** *The combinations are chosen on the basis of the underlying theory of the data (e.g. numerical analysis)*

Questions of Strategy

- *Is your testing driven by / drivable by an oracle?*
- *Is your objective high-volume or hand-crafted?*
- *Is your combinatorial direction “space” or “time”?*
 - *Time: We watch the program’s behaviour over time: sequence analysis, sequential circuit testing, state models*
 - *Space: We watch the program’s behaviour as we set values for different variables or set relationships for groups of variables: boolean expressions (multi), combination circuit testing, cause-effect graphing*
- *Are you looking for bugs that are primarily combinational or are you expecting to expose many single-factor bugs (bugs that could have been found with single-variable tests but that you are reaching them through the combinations)?*
- *Are you constrained or enabled by the characteristics of the tester(s)?*

Questions of Strategy

- *Are you looking for accidental combinations of should-be-independent variables or studying interactions of should-be-related variables?*
- *Are you looking for failures that are obviously tied to a specific combination or failures that emerge over time as a result of side-effects of specific combinations?*
 - *Memory leaks, stack overflow, etc.*
- *Is timing a variable?*
 - *Multi-processor systems, extent of monitoring or diagnostics*
- *How will you recognize a failure?*
 - *Comparison oracle, diagnostics, obvious-to-the-tester, model*

Strategy:

Practicalities of Key Value Selection

For a given variable, how will you decide which values to select?

- *Domain testing (classical—identify equivalence classes and boundaries)*
- *Domain testing (risk-based—identify equivalence classes based on theories or guesses of error and pick best representatives)*
 - *For example, deep understanding of non-ordered sets (like printers)*
- *Data-mined analysis*
 - *Tech support data*
 - *User interviews*
- *Market analysis*
 - *Expected configuration characteristics*
 - *Common application characteristics (e.g. common employee compensation plans)*