The sareport class*

Laurens Sion
laurens.sion@cs.kuleuven.be

February 6, 2019

Abstract

The `sareport` class is an extended version of the standard LaTeX `report` class with several of the most commonly needed packages already loaded and providing a uniform layout for all the student reports for the *Software Architecture* course.

Contents

*This document corresponds to sareport v1.37, dated 2019/02/06.

1 Introduction

The `sareport` class is an extended version of the standard LaTeX `report` class with several of the most commonly needed packages already loaded and providing a uniform layout for all the student reports for the *Software Architecture* course. It provides a range of information messages, hints and warnings to help you fill in the report. It also provides some configuration options to set your team information.

2 Usage

2.1 Setting cover sheet information

The following information can be set to customize the first page with your team information.

2.1.1 Authors & group name

Use the following commands to set the author information:

`\addAuthor` Usage: `\addAuthor{`⟨*First Name*⟩`}{`⟨*Last Name*⟩`} {`⟨*Student Number*⟩`}`.
Use this macro to add an author to the report. Don't forget the student number as well. The student number must be unique. You can add up to three authors using this command. The groupname will be automatically created based on the added names.

`\groupname` Usage: `\groupname{`⟨*ALastname-BLastname-ZLastname*⟩`}`.
Use this macro to set the groupname. Make sure the lastnames are sorted alphabetically and any spaces are removed. Normally `sareport` can automatically determine your groupname. You *only* need to manually override this when the automatically determined group name is incorrect. if you manually override the name, you must do so after setting the authors.

2.1.2 Course info

Use the following command to set the course information.

`\academicyear` Usage: `\academicyear{`⟨*2018 -- 2019*⟩`}` .
Use this macro to set the academic year. You can omit this, in that case, the year will be automatically generated.

`\casename` Usage: `\casename{`⟨*name*⟩`}` .
Use this macro to set the name of the case. This name will be provided to you.

`\phasenumber` Usage: `\phasenumber{`⟨*number*⟩`}` .
Use this macro to set the number of the phase.

`\phasename` Usage: `\phasename{`⟨*name*⟩`}` .
Use this macro to set the name of the phase. This name will be provided to you.

`\course` Usage: `\course{`⟨*name*⟩`}`.
Use this macro to customize the course name in the bottom left corner.

`\department` Usage: `\department{`⟨*name*⟩`}`.
Use this macro to customize the department name under the KU Leuven logo. The size can be changed if the text does not fit nicely e.g., `\department{\small text}`.

2.2 Setting or removing instructions

The instruction commands listed below are provided. They are used in the template to provide helpful information when filling in the report. Make sure to remove these before submitting the final version of the report.

`\captioninstruction` Instructions on how to set the captions for your figures.

`\showdecisionsnotes` Informative notes for the architectural decisions.

`\showcsnotes` Informative notes for the client-server view.

`\showdecompnotes` Informative notes for the decompositions.

`\showdeploynotes` Informative notes for the deployment.

`\showscenariosnotes` Informative notes for the scenarios.

`\showcatalognotes` Informative notes for the element catalog.

2.3 Todo notes

The following commands are used internally for showing the instructions and notes mentioned above. They are available for you to use in your project as well.

`\todoinline` Usage: `\todoinline[`⟨*todoargs*⟩`]{`⟨*text*⟩`}`. The `todoinline` macro has a single mandatory argument for the ⟨*text*⟩ that needs to be displayed in the todonote. You can also pass optional arguments (⟨*todoargs*⟩) to the underlying `\todo`-command of the `todonotes` package.

`\note` Usage: `\note[`⟨*todoargs*⟩`]{`⟨*text*⟩`}`. The `note` macro has a single mandatory argument for the ⟨*text*⟩ that needs to be displayed in the todonote. You can also pass optional arguments (⟨*todoargs*⟩) to the underlying `\todo`-command of the `todonotes` package.

`\hint` Usage: `\hint[`⟨*todoargs*⟩`]{`⟨*text*⟩`}`. The `hint` macro has a single mandatory argument for the ⟨*text*⟩ that needs to be displayed in the todonote. You can also pass optional arguments (⟨*todoargs*⟩) to the underlying `\todo`-command of the `todonotes` package.

`\warning` Usage: `\warning[`⟨*todoargs*⟩`]{`⟨*text*⟩`}`. The `warning` macro has a single mandatory argument for the ⟨*text*⟩ that needs to be displayed in the todonote. You can also pass optional arguments (⟨*todoargs*⟩) to the underlying `\todo`-command of the `todonotes` package.

If your report still contains any notes or instructions (using any of the above four commands), the following warning is printed on the titlepage:

> ⚠ **Attention:** Your report still contains instructions and comments. Make sure to delete all of them for the final version. (Run `pdflatex` at least two times after removing them to make sure this warning disappears.)

2.4 The element catalog

There are 2 commands you can use when you want to manually construct your element catalog. If you want to use the exported version of the plugin, just add `\input{exported_catalog}` at the end.

`\componentItem` Usage: `\componentItem{`⟨*name*⟩`}{`⟨*contents*⟩`}`. Specify the all the information about a single component. Components are sorted using the ⟨*name*⟩ argument. Specify the responsibility, interfaces and operations in the ⟨*contents*⟩ argument.

`\printComponents` Usage: `\printComponents`. Print the sorted list of all the components and their interfaces and operations.

3 Problems, Issues, and Questions

For problems, issues, or questions with this class contact the `SA Team` at:

`SoftwareArch2019@cs.kuleuven.be`