Component-Based Software Engineering	**Exercise Sheet No. 4**
Dipl.-Medieninf. Christian Piechnick	Software Technology Group
INF 2078	Institute for SMT
christian.piechnick@tu-dresden.de	Department of Computer Science
	Technische Universität Dresden
	01062 Dresden

Aspect-Oriented Programming

Task 1: Aspect-oriented Programming (AOP)

1a) Describe the terms *Scattering* and *Tangling*. Why are both effects bad?

1b) What does cross-cutting mean w.r.t. AOP?

1c) Describe the terms *Pointcut*, *Join Point*, *Advice* and *Weaving*. Give an example.

1d) Where is the differenc between static and dynamic Join Points?

1e) Where is the difference between static and dynamic weaving?

1f) Describe the component model, composition technique and composition language of AspectJ.

Task 2: Generic Programming with Generic Components

2a) What is a generic component?

2b) What makes a language *fully generic*? What is consequence?

2c) What is a hook? What is slot? Where is the difference?

2d) Compare Aspect-oriented programming and Generic Programming with Generic Components? Where are the commonalities and differentialities?

Task 3: AspectJ

In this task your are using AspectJ to write some simple aspects and learn how aspect-oriented software is implemented. AspectJ is a aspect-oriented extension to java. For eclipse there are the AspectJ Development tools available.
We will provide advanced features for our production system using aspects. The solution of the previous task will serve as the base application which we will extend using aspects.

1. Install the AspectJ Development Tools AJDT. (Choose the right update site, according your Eclipse version: `http://www.eclipse.org/ajdt/downloads/index.php`)
2. Download the source of the core project. (from the CBSE website)
3. Edit the core project to solve the task (add the aspects).

AJDT Website: `http://www.eclipse.org/ajdt/`
AspectJ Documentation: `https://eclipse.org/aspectj/doc/released/progguide/starting.html`
AspectJ Getting Started Tutorials: `http://o7planning.org/web/fe/default/en/document/18642/java-aspect-oriented-programming-tutorial-with-aspectj`

3a) Define an Aspect that logs (prints to the console) when the programm starts and stops running.

3b) Define an Aspect which restricts that levels cannot contain other levels.

3c) Define an Aspect which marks a bathroom as not clean, when it is entered.

3d) Define an Aspect which prints a warning when a room is entered which is not cleaned.

3e) Define an Aspect which prohibits (e.g., throws an exception) any person entering a bathroom which is not cleaned.

3f) In the initialization Script (buildSimpleHouse()), a house with a private room is created. Define an aspects which prohibits any other person that the owner of the house, to enter the private room.