Design Patterns and Frameworks	**Exercise Sheet No. 10**
Dipl.-Inf. Steffen Zschaler	Software Engineering Group
INF 2097	Institute for Software and Multimedia Technology
`http://st.inf.tu-dresden.de/teaching/dpf`	Department of Computer Science
	Technische Universität Dresden
	01062 Dresden

Frameworks

In this exercise we will look at a few frameworks and analyse them for template hook patterns.

Task 1: Hook Fundamentals

1a) **Task:**

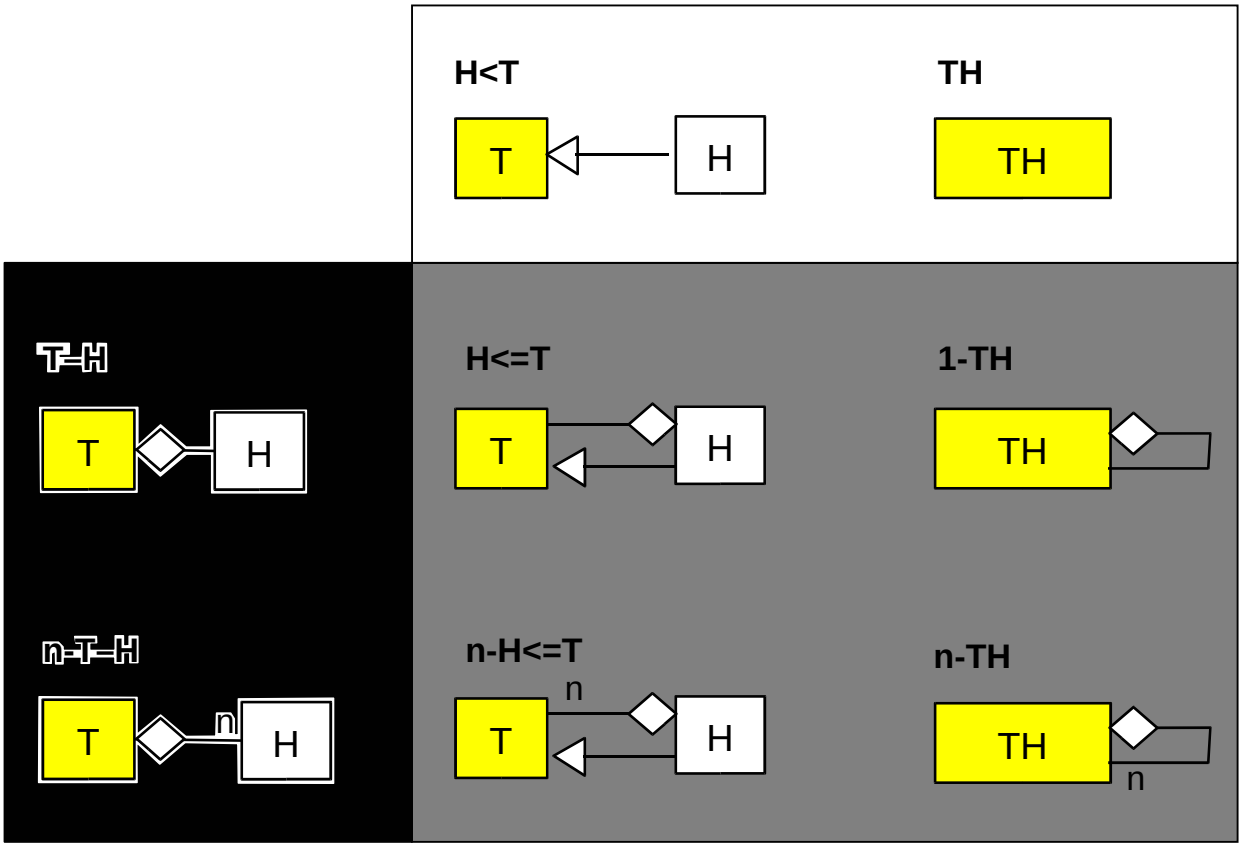
What framework hook patterns do you know?

Solution: See next sub-task.

1b) **Task:**

Which of these patterns are typical for black-box reuse? Which are typical for white-box reuse?

Solution:



Task 2: Log4J

Log4J is a framework supporting powerful logging statements in Java programs. Check out the framework at http://logging.apache.org/log4j/docs/ and understand the core principles.

2a) **Task:**

Look at the classes `Appender` and `Layout` and their relationship. What template–hook pattern can you find here?

Solution: There is a *T*—*H* pattern with `Appender` as the template and `Layout` as the hook.

2b) **Task:**

Look at the classes `Logger` and `Appender` and their relationship. What template–hook pattern can you find here?

Solution: There is a *nT*—*H* pattern with `Logger` as the template and `Appender` as the hook.

2c) **Task:**

From the hooks you found: What kind of framework is Log4J? Is it rather black-box or white-box?

Solution: The hooks are typically of the *xT*–*H* variety; that is they are hooks typical of black-box frameworks.

Task 3: JUnit

JUnit is a framework for unit-testing Java programs in a manner supporting regression tests. Check out the framework at http://junit.org/index.htm and understand the core principles.

3a) **Task:**

Look at the `TestCase` class. What template–hook pattern can you find here?

Solution: There is a *TH* pattern (equivalent to *H*<*T*) here. The `run` operation constitutes the template part; `setup`, `runTest`, and `tearDown` are realised in the hook.

3b) **Task:**

Look at the classes `TestSuite` and `Test` and their relationship. What template–hook pattern can you find here?

Solution: There is a *nT*—*H* pattern with `TestSuite` as the template and `Test` as the hook. The interesting bit is that `TestSuite` is a subclass of `Test`, which could mislead one to interpret the hook as a *nH*<=*T* hook. However, because tests are what you vary, `Test` is the hook, and the pattern is *nT*—*H*.

3c) **Task:**

Look at the classes `TestCase` and `TestResult` and their relationship. What template–hook pattern can you find here?

Solution: There is no pattern here at all, because there is no variation point.

3d) **Task:**

From the hooks you found: What kind of framework is JUnit? Is it rather black-box or white-box?

Solution: The hooks are typically of the *xT–H* variety; that is they are hooks typical of black-box frameworks. However, one of the core hooks (`TestCase`) is a *TH* hook, which is the archetypical white-box hook. Therefore, I would say that the framework is somewhere between black and white, but leans more to the white side.