

# **The Second Mandatory Programming Assignment**

## **Part 2:**

### **Zookeeper concepts**

*INF5040/INF9040 Autumn 2017*

#### **Objective**

- To understand reliable distributed coordination concepts in Zookeeper.
- To get familiar with the technique of watches in order that is used as a building block in many solutions for various distributed coordination problems.

#### **Scope**

Zookeeper is a centralized service for maintaining configuration information, naming, providing distributed synchronization and group services. For this part of the assignment, you need to use the standard client shell to connect to a Zookeeper server. Please follow the instructions in order to produce correct results while answering the questions.

#### **Development Tools:**

1. *Integrated Development Environment:* Eclipse IDE for **Java EE** Developers:  
<http://www.eclipse.org/downloads/packages/eclipse-ide-java-ee-developers/junor>
2. Zookeeper server and client shell:  
<http://zookeeper.apache.org/releases.html>

#### **Deliverable:**

- A compressed file (zip) containing:
  - A document in the PDF format that contains answers to the questions below with screenshots illustrating the results you get.

**Deadline: November 6, 2017, 23:59**

**How to deliver?** Via the Devilry system.

# ***Zookeeper Laboratory instructions***

## **1. Ephemeral vs Persistent (Regular) znodes**

### **1.1 Create a znode for this exercise /<your group id>/ex1**

Example:

```
[Zookeeper Command line] create /<your group id> data
```

All znodes that we created so far are regular zNodes.

### **1.2 (Create Persistent) Create another persistent node (e.g., /<your group id>/ex1/persistent)**

### **1.3 (Create Ephemeral) Now create one ephemeral znode (e.g, /<your group id>/ex1/ephemeral) using create command with option -e between create and the path (create -e <path> <data>).**

### **1.4 (List/getChildren) List the children of /<your group id>/ex1 using ls command (Note that ls in the zkCli.sh corresponds to getChildren in the API)**

### **1.5 (Test Ephemeral) Now quit the client by typing**

```
[Zookeeper Command line] quit
```

Then restart the client and list the children of /<your group id>/ex1 again using ls command.

Are the results in 1.5 and 1.4 different? Explain your observation.

### **1.6 (Ephemeral children) Create a new ephemeral znode (e.g, /<your group id>/ex1/ephemeral1) and try to create a child of this ephemeral node (e.g, /<your group id>/ex1/ephemeral1/child). What happens?**

## 2. Sequential suffix

### 2.1 Create the znode for this exercise /<your group id>/ex2

### 2.2 (Sequential basics) Create a few znodes with SEQUENTIAL suffix (simply called sequential znodes) as children of /<your group id>/ex2. Try option -s in create

[Zookeeper Command line] `create -s /<your group id>/ex2/child someData`

Answering the following questions:

- Q1: How is the name of a sequential znode constructed? How long is the suffix?
- Q2: Is it possible to create persistent sequential znodes?
- Q3: Is it possible to create ephemeral sequential znodes?
- Q4: Now create a few non-sequential znodes and then delete them. Create a few sequential znodes. Explain how the suffix numbers are generated.
- Q5: Assume that right after creating /<your group id>/ex2, you created 20 sequential children of /<your group id>/ex2, followed by 5 non-sequential children of /<your group id>/ex2, followed by deleting 3 most recent sequential child znodes. If you'd create another sequential node, what would be its suffix?

### 2.3 (Scope of sequence numbers) Are sequence numbers of sequential children of a znode (e.g, /<your login id>/ex2) and sequence numbers of sequential children of one of ex2 znode's children (e.g, /<your login id>/ex2/child1) related?

### 2.4 (Sequence numbers across multiple clients) Are sequence numbers unique to a given client? To answer this question, you may want to start two (or more) clients in separate windows (or on separate machines), both creating sequential znodes as children of /<your login id>/ex2

### 3. Watches

All read operations in Zookeeper have the option to set a watch as a side effect. The goal of this exercise is to understand the behavior of watches in Zookeeper and give you ideas on how watches can be used in coordinating distributed systems.

#### 3.1 (Init) Create the root znode for this exercise with path

<root\_ex3> = /<your login id>/ex3

Start the second client in parallel. In one client, you will set the watches while in the other, you will make actions that might trigger the watches.

#### 3.2 (First watch)

In client 1, do a get on path <root\_ex3> and set a watch.

[Zookeeper Command line] get /zookeeper/ex3 true

Then, in Client 2, modify data for <root\_ex3>

[Zookeeper Command line] set /zookeeper/ex3 newData

What happens in Client 1?

#### 3.3 (Durability of watches) Now modify again <root\_ex3> in Client 2. What happens in Client 1? How many times can a watch be triggered in Zookeeper?

#### 3.4 getChildren command can also set a watch. Do this in Client 1 on path <root\_ex3> (Remember getChildren corresponds to ls in the client shell). Now, modify again the data on <root\_ex3> in Client2. Does this trigger the watch at Client 1? Explain why.

#### 3.5 (Child watches) In Client 2 create a child node of <root\_ex3>. What happens in Client 1? Create another child of <root\_ex3> in Client 2. What happens in Client 1?

#### 3.6 (Summarizing watch types) Explain the difference between child watches and data watches. Which command sets which watch?

#### 3.7 (Child watches and nesting) In this exercise we test nested child watches.

In Client 1, create another child of <root\_ex3>. Set child watches on both newly created child and <root\_ex3>.

Now, in Client 2, create a grandchild (a child of a child) of <root\_ex3>. How many watches are triggered in Client 1?

Create yet another child of <root\_ex3> in Client 2. Does this trigger the watch in Client 1? Do grandchildren trigger child watches in Zookeeper?

#### 3.8 (Watches and znode deletion) Create a node /<your group id>/ex3/deletetest

Run three clients in parallel:

- In Client 1, set a child watch on deletetest znode.
- In Client 2, set a data watch on deletetest znode.
- Finally, in Client 3, delete the deletetest znode.

Which watches are triggered by a delete?

#### 3.9 More on watches:

- Watches are maintained locally in the Zookeeper server to which the client is connected.

- The order of watch events from Zookeeper corresponds to the order of the updates seen by the Zookeeper service.
- Due to asynchrony and network delays, different clients may see watches triggered at different times. The time when a watch triggers cannot be used as synchronization method itself.