If you're developing Zimbra extensions and have perhaps read my post about **Zimbra Server debugging**, you might be wondering about how to use logging in your extension.

Zimbra uses **log4j** for logging purposes – which is a quite common and comfortable way. Additionally, Zimbra offers a proxy class to handle logging for you. I will not tell you about things like severity levels here. You should have a fairly good idea about what logging is anyways.

To better separate logging for different areas, Zimbra has implemented a category system for logs. You can see the available categories in the **Zimbra Wiki**. So, embrace that category system and use the best category, when logging something.

The way to do **that** is by just statically calling the ZimbraLog utility class with the specific category, the severity, log message and maybe additional objects to print out. For example:

```
ZimbraLog.security.debug("User %s tries to authenticate.", acct);
```

As you can see, you can also use format signs (%s) to embed the provided objects into the log message. (There are actually a lot of message formats available. Check the **log4j documentation** for details)

To be able to actually **see** the log messages somewhere, you have multiple options:

- Add an account logger by calling
  `zmprov addAccountLogger <account> <category> <severity>`
  Beware, though, that an account logger may not be available while your extension is called, as it is only availble in an account context (for example, when you're logged in)
- Configure logging without restarting the mailboxd by editing the file **~zimbra/conf/log4j.properties** and afterwards calling
  `zmprov resetAllLoggers`
  Beware, though, that all changes are removed once the *mailboxd* is restarted.
- Configure a fixed logging setting by editing the file **~zimbra/conf/log4j.properties.in** and restarting the *mailboxd* by calling
  `zmmailboxdctl restart`

The files **log4j.properties{.in}** expect the following entry format:

```
log4j.logger.zimbra.<category>=<severity>,<appender>
```

The „<appender>" parameter relates to a appender-configuration, which is also configured

in the same file and is used to writes log messages to specific files. For example, the „*LOGFILE*„-appender writes log messages to the file **~zimbra/log/mailbox.log**, while the „*AUDIT*„-appender uses the file **~zimbra/log/audit.log**.

While this is a bit complicated, it may help you determine the problem in your code without having to set up the debugging environment, which could be a bit nasty in a production environment.