参考

http://knowledge.reontosanta.com/archives/648

http://blog.livedoor.jp/applepedlar/archives/53526919.html

java.util.logging.Logger

ログを出力する

```
// Logger クラスのインスタンスを生成する
final Logger logger = Logger.getLogger("SampleLogging");
// ログを出力する
logger.log(Level.FINEST, "ログ出力テスト：finest");
logger.log(Level.FINER, "ログ出力テスト：finer");
logger.log(Level.FINE, "ログ出力テスト：fine");
logger.log(Level.CONFIG, "ログ出力テスト：config");
logger.log(Level.INFO, "ログ出力テスト：info");
logger.log(Level.WARNING, "ログ出力テスト：warning");
logger.log(Level.SEVERE, "ログ出力テスト：severe");
```

ファイルに出力する

```
// Logger クラスのインスタンスを生成する
final Logger logger = Logger.getLogger("SampleLogging");
try {
    // 出力ファイルを指定する
    FileHandler fh = new FileHandler("SampleLog.log");
    // 出力フォーマットを指定する
    fh.setFormatter(new java.util.logging.SimpleFormatter());
    logger.addHandler(fh);
} catch (IOException e) {
    e.printStackTrace();
}
// 出力レベルを CONFIG 以上に設定する
logger.setLevel(Level.ALL);

// ログを出力する
logger.log(Level.FINEST, "ログ出力テスト：finest");
logger.log(Level.FINER, "ログ出力テスト：finer");
logger.log(Level.FINE, "ログ出力テスト：fine");
logger.log(Level.CONFIG, "ログ出力テスト：config");
logger.log(Level.INFO, "ログ出力テスト：info");
logger.log(Level.WARNING, "ログ出力テスト：warning");
logger.log(Level.SEVERE, "ログ出力テスト：severe");
```

標準出力に出さないようにする

ファイルに出力しても、標準出力される。
標準出力に出力しないようにするには、親 Logger の Handler から ConsoleHandler を除去する必要がある

```
public class Test {
  public static void main(String[] args) throws Exception {
    Logger logger = Logger.getLogger("Test");
    FileHandler fileHandler = new FileHandler("testlog.txt");
    fileHandler.setFormatter(new SimpleFormatter());
    logger.addHandler(fileHandler);
    logger.log(Level.INFO, "Test1 コンソール表示あり ");
    removeConsoleHandler(logger);
    logger.log(Level.INFO, "Test2 コンソール表示なし ");
  }
  private static void removeConsoleHandler(Logger logger) {
    Handler[] handlerArr = logger.getHandlers();
    for (Handler handler : handlerArr) {
      if (handler instanceof ConsoleHandler) {
        logger.removeHandler(handler);
      }
    }
```

```
        // 再帰
        Logger parentLogger = logger.getParent();
        if (parentLogger != null) {
          removeConsoleHandler(parentLogger);
        }
      }
    }
```

その他

properties ファイルの内容は、Logger の最上位の親に適用される。

java.util.logging.config.file で指定した Handler や Level は

```
Logger.getLogger("global").getParent().getHandlers();
Logger.getLogger("global").getParent().getLevel();
```

などで取得できる。

API ドキュメントより
各 Logger は、Logger の名前空間にある既存の上位クラスにもっとも近い「親」Logger を追跡します。

各 Logger は各々に関連した Level を持ちます。これは、このロガーが対象とする最小 Level を反映します。Logger のレベルが null に設定される場合、Logger の有効なレベルはその親から継承され、そしてまた親はその親から再帰的に有効なレベルを取得します。これは、ツリーの上位方向に以下同様に行われます。

（略）
デフォルトでは、ロガーは親の Handler にも通知します。これは階層の上位方向に再帰的に行われます。