# Thread Group Syntax JMeter

In the `CurrentThreadInfo` class, the methods used are part of the `Thread` class in Java. Here's a detailed explanation of each method and what it does:

1. `currentThread()`

   - **Usage**: `Thread.currentThread()`
   - **Purpose**: This is a static method of the `Thread` class that returns a reference to the currently executing thread. It allows you to get information about the thread that is running the code.

2. `getName()`

   - **Usage**: `currentThread.getName()`
   - **Purpose**: Returns the name of the thread. Every thread has a name, and by default, the main thread is called `"main"`. You can set or change the name of a thread with the `setName(String name)` method.

3. `getId()`

   - **Usage**: `currentThread.getId()`
   - **Purpose**: Returns the unique identifier (ID) for the thread. This ID is assigned by the JVM when the thread is created and remains the same throughout the thread's lifetime.

4. `getPriority()`

   - **Usage**: `currentThread.getPriority()`
   - **Purpose**: Returns the priority of the thread as an integer between 1 (minimum) and 10 (maximum). Thread priority determines how the operating system schedules the thread relative to others. The default priority is `5`.

5. `getState()`

   - **Usage**: `currentThread.getState()`
   - **Purpose**: Returns the current state of the thread as an enumeration (`Thread.State`). A thread can be in one of the following states:
     - `NEW`: The thread has been created but not yet started.
     - `RUNNABLE`: The thread is running or ready to run.
     - `BLOCKED`: The thread is blocked waiting for a monitor lock.
     - `WAITING`: The thread is waiting indefinitely for another thread to perform an action.
     - `TIMED_WAITING`: The thread is waiting for a specific period of time.
     - `TERMINATED`: The thread has exited.

6. `isDaemon()`

   - **Usage**: `currentThread.isDaemon()`
   - **Purpose**: Checks if the thread is a daemon thread. A daemon thread is a background thread that

terminate, even if daemon threads are still running. The main thread is **not** a daemon by default, but you can set other threads to be daemon threads using `setDaemon(true)`.

### 7. `isAlive()`

- **Usage**: `currentThread.isAlive()`
- **Purpose**: Returns `true` if the thread is alive, meaning it has been started and has not yet terminated. This is useful to check if the thread is still running.

### Summary of Methods:

- `Thread.currentThread()`: Returns the current thread.
- `getName()`: Returns the thread's name.
- `getId()`: Returns the unique thread ID.
- `getPriority()`: Returns the thread's priority.
- `getState()`: Returns the current state of the thread.
- `isDaemon()`: Checks if the thread is a daemon thread.
- `isAlive()`: Checks if the thread is still running.

This program prints out detailed information about the thread currently executing the `main()` method, which is typically the "main" thread in any Java program.