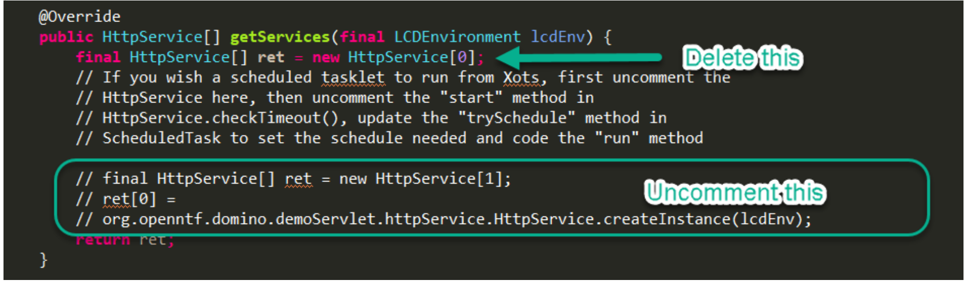
HTTPServices

Version 2.1.0 includes an HTTPServiceFactory and the functionality to add a sceduled task. However, this is not enabled by default.

In the plugin's plugin.xml there is an extension to `com.ibm.commons.Extension` to add a service factory. It points to a class called HttpServiceFactory in the plugin's package ending "httpService" (the full package name will vary depending on what you call your plugin). If you open that class you will see the steps to enable the HttpService:

1. Uncomment the lines near the bottom of the class. The first line of the comment declares the HttpService array variable `ret` with a size of 1, so you will need to delete the line that declares it with a size of 0.

   ```
   @Override
   public HttpService[] getServices(final LCDEnvironment lcdEnv) {
       final HttpService[] ret = new HttpService[0];
       // If you wish a scheduled tasklet to run from Xots, first uncomment the
       // HttpService here, then uncomment the "start" method in
       // HttpService.checkTimeout(), update the "trySchedule" method in
       // ScheduledTask to set the schedule needed and code the "run" method

       // final HttpService[] ret = new HttpService[1];
       // ret[0] =
       // org.openntf.domino.demoServlet.httpService.HttpService.createInstance(lcdEnv);
       return ret;
   }
   ```

2. In the HttpService class, uncomment the line in the `checkTimeout{}` method. That's commented out currently as an extra safety-net.
3. The `start()` method of the `ScheduledTask` Xots tasklet calls the `trySchedule{}` method. This creates a `PeriodicScheduler` to determine how frequently the tasklet should run and when it should start. This will probably need amending. It then schedules the tasklet
4. Amend the `run()` method to do what you need to, just like in a normal Xots tasklet.

The HTTPService then gets loaded when HTTP starts. The `getPriority()` method determines what priority each HTTPService has within the runtime. The `checkTimeout` method of each HTTPService then runs every 30 seconds. The part that will be uncommented is `ScheduledTask.getInstance().start();`. This ensure only one instance of the ScheduledTask class is instantiated and the `start()` method checks whether it's already been started, so ensures the tasklet is only added to Xots once (this happens in the `trySchedule()` method).

The `trySchedule` method creates a `PeriodicScheduler` with a setting to run daily with no initial delay. But the `eventStart` of the scheduler is set to 4am the next morning. So this tasklet will not run until 4am the next day and then daily at 4am. If you wanted it to not run at weekends, you could set an `eventEnd` and then, on the last run, schedule the tasklet with a new `eventStart` of Monday morning and a subsequent `eventEnd`. (If possible, I think it would be better to update the schedule for the current tasklet. But I'm not sure if that's possible. Otherwise the tasklet may sit SLEEPING after the `eventEnd`.)

Once added to Xots, the `run()` method gets triggered. You just need to add your code into that method.