Job Scheduler Overview

In this codelab, I learned how to implement JobService to an application, how to construct a JobInfo object with specific constraints, and how to schedule a JobService based on the JobInfo object. JobScheduler provides a flexible framework to accomplish background services. The JobScheduler class allows you to set the conditions, or parameters for when to run your task. Jobscheduler will then calculate the best time to schedule the execution of the job. Each task to be run is implemented as a JobService subclass and executed as per the specified parameters. The JobScheduler class is available only for devices running on API 21 and higher. For backward compatibility, use the WorkManager API which allows you to schedule background tasks.

In the JobScheduler codelab, the app demonstrates by allowing the user to select constraints and schedule a job. In order to use a JobScheduler, one would need to use:

* A JobInfo object, which contains the set of conditions that trigger a job to run.
* A JobService, which is the implementation of the job that runs under the conditions set from the JobInfo object.

To implement JobScheduler, we have to begin by:

1. Implement a JobService that will run at a time set by the conditions. The following need to be implemented:
   1. onStartJob() callback. This callback is called when the system determines that your task should be run. You would need to implement the job to be done in this method. This method returns a boolean to indicate whether the job needs to continue on a separate thread. If true is returned, jobFinished() must be manually called and the work is offloaded to a different thread. If false, the system will automatically call jobFinished() on your behalf.
   2. onStopJob() callback. This callback is called if the conditions described in the JobInfo are no longer met, the job must be stopped and the system calls onStopJob().
2. Implement the job conditions (JobInfo). In this step, you must create a series of parameterized conditions for running a job.

**Answer these questions**

**Question 1**

What class do you use if you want features like the ones provided by JobScheduler, but you want the features to work for devices running API level 20 and lower?

* JobSchedulerCompat
* **WorkManager**
* AlarmManager