Decoupling Work Using Queues & Scheduled Tasks

Google App Engine

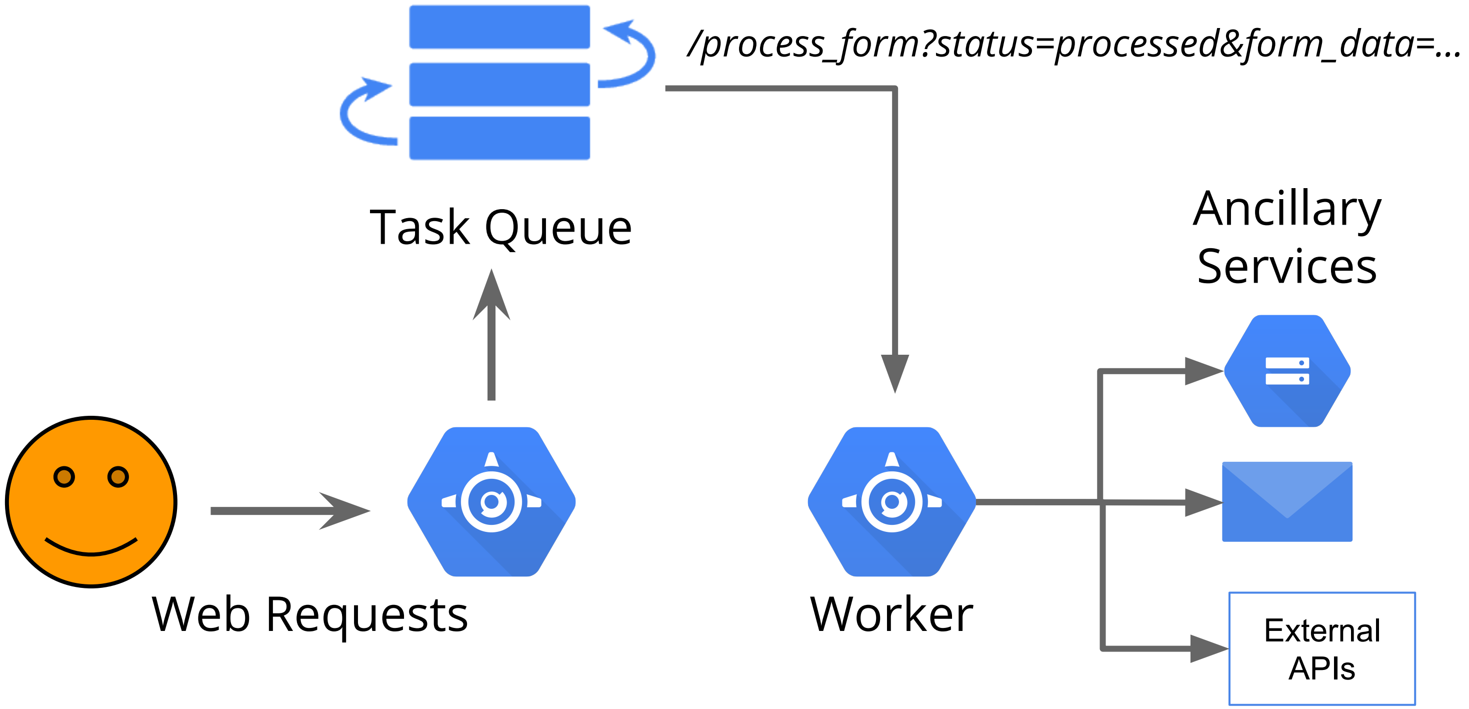
Google Cloud Platform

Agenda

Task Queues

- Task is a unit of work to be executed by your application or external workers.
    - Executed asynchronously outside a user request
    - Use Cases:
        - Sending e-mail
        - Batching datastore writes
- Task Queue is an App Engine service to manage and run tasks.

Task Queues


/process_form?status=processed&form_data=...
Task Queue
Ancillary Services
Web Requests
Worker
External APIs

Two Kinds of Task Queue Services

Push queues

- App Engine managed service
  - App Engine schedule/execute/delete/retry automatically
  - Your application creates tasks with HTTP request handlers
- Less control and only directly accessible from your App Engine app

Pull queues

- Manual service
  - Your application schedules, executes, deletes, and retries task
- More control and available from outside App Engine app

How to Configure Queues?

Configure queues in a queue config file

- Java: **`queue.xml`**
- Python: **`queue.yaml`**

In the queue config file, you can specify

- Total storage limit (Optional safety check)
- For each queue,
  - Queue name (Required)
  - Mode (Push/Pull)
  - Other parameters specific to each queue type

config docs

- Java
- Python

Queue Configuration Files

Java

queue.xml

```
<queue-entries>
  <queue>
    <name>push-queue-1</name>
    <rate>1/s</rate>
  </queue>
  <queue>
    <name>pull-queue-1</name>
    <mode>pull</mode>
  </queue>
</queue-entries>
```

Python

queue.yaml

```
queue:
- name: push-queue-1
  rate: 1/s
- name: pull-queue-1
  mode: pull
```

Adding A Task to A Queue

1. Get queue
2. Create task with options
3. Add task to queue

Java

```java
Queue queue = QueueFactory.getQueue("Qname");
TaskOptions task = TaskOptions.Builder(...)
queue.add(task);
```

Python

```python
queue = taskqueue.Queue("Qname")
task = taskqueue.Task(...)
queue.add(task)
```

Adding Tasks in A Datastore Transaction

- Tasks can be enqueued during a transaction
- Task does not *execute* within the transaction

Java

```
Transaction txn = ds.beginTransaction();
queue.add(TaskOptions.Builder.withUrl("/handler"));
txn.commit();
```

Python

```
def a_method(parameter…):
  taskqueue.add(url='/handler', transactional=True)

db.run_in_transaction(a_method, parameters...)
```

Push Queues

Google Cloud Platform

Push Queue Overview

- Task Definition
  - Name, parameters, headers, and payload etc. with max 100KB task size
  - Specify handler (internal URLs) and method (GET/POST)
- Task Scheduling Per Queue
  - Executed ASAP and as many as possible with *token bucket algorithm*
  - Processed in FIFO order (mostly), ETA or countdown
- Task Execution:
  - Tasks processed as HTTP requests (GET/POST)
  - May cause new instances to be created
  - 10 minute limit for request processing for automatic scaling
  - 24 hour limit for request processing for basic and manual scaling
  - Task is deleted after successful execution, back to queue if fails

Push Queue Task Definition

Specify options in code when creating tasks

- url
- task name
- payload
- params
- headers
- countdown / eta
- target
- retryOptions

- TaskOptions JavaDoc
- Task Reference Python (*includes nice summary of push vs pull options)*

Order of Task Execution

Order of task execution is best effort based on

- Position of task in the queue (FIFO)
- Task's eta or countdown property
  - Mutually exclusive
  - Task is executed on or after eta or countdown
- Backlog of tasks in the queue
  - New task might queue jump to optimize execution in case of big backlog

Default Queue

- App Engine has a default push queue
- No need to configure it—uses default settings
- If you want to configure it:
    - Define queue named default in queue config file

Java

```
Queue queue = QueueFactory.getDefaultQueue();
Queue queue = QueueFactory.getQueue("Qname");
```

Python

```
queue = taskqueue.Queue()
queue = taskqueue.Queue("Qname")
```

Tuning Task Scheduling

Parameters for token bucket algorithm:

- rate -- usual rate
- bucket-size -- cap on peak demand
- max concurrent requests

Example:

- rate = 10
- bucket-size = 40

Queue can execute 10 tasks per second. If rate is not used up, accrue up to max of 40 tasks.

What Happens if a Task Fails?

- HTTP 200 - 299 response code for success; otherwise fails.
- If a task fails, it goes back in the queue
- App Engine retries it until it succeeds
- Specify task retry options
    - In queue config file
    - In code when creating the task

Task Retry Options

- Retry Limits
    - **`task-retry-limit`** - min retries
    - **`task-age-limit`** - min elapsed time to keep retrying
        - If both task-retry - limit and task-age-limit are reached, task is deleted
- Backoff Strategy
    - **`min-backoff-seconds`** -- min amount of time to wait before the first retry
    - **`max-backoff-seconds`** -- max delay between retries
    - **`max-doublings`** -- max times to double delay

Push Queue in Development Server

- The dev server executes tasks at the appropriate time
- Executes tasks as close to their ETA as possible

However

- Does not respect rate and bucket-size
- Setting a rate of 0 does not prevent tasks from being executed automatically
- Does not retry tasks
- Does not preserve queue state across server restarts

Push Task Request Headers

- `X-AppEngine-QueueName`
- `X-AppEngine-TaskName`
- `X-AppEngine-TaskRetryCount`
- `X-AppEngine-ExecutionCount`
- `X-AppEngine-TaskETA`
- `X-AppEngine-FailFast:` failing without spawning new instances

These headers are set internally by App Engine. Stripped out if request comes from outside App Engine.

Push Queue Best Practices

- Set security so only admins can access handler URLs
- Use different queues for different purposes
    - Allows rates to be fine-tuned for performance/cost trade-off
- Batch task creation (< 100 tasks per batch)
- If order is critical, implement a control mechanism
- Specify target to use a specific version
    - Unless specified otherwise, task executes on the version of the app that spawned it

Pull Queues

Google Cloud Platform

Pull Queue Overview

- Task Definition (< 1MB)
  - Do not specify handler, target, method (always PULL)
  - Tasks can have tags; you can lease tasks by tag
- Task Queue Definition
  - Set mode to PULL
  - No default pull queue
- Task Execution
  - Task leased by worker - no scheduling, no auto retry
  - Workers must delete tasks -- no auto deletion
- REST interface (with ACL)

Pull Queue Task Options

- task name
- method (PULL)
- payload
- params
- tags (Beta)
- task retry limit

- TaskOptions JavaDoc
- Task Reference Python (*includes nice summary of push vs pull options)*

Processing Pull Tasks

- Lease Tasks
  - Workers lease tasks from the queue for a period of time
  - Lease up to 1000 tasks at a time
  - Modify lease to extend duration
- Delete Tasks
  - Worker must explicitly delete task when done
  - If task is not deleted when lease expires, it goes back in the queue
- Retry Tasks = Lease Tasks Again

Pull Queue Best Practices

- Scale up and down workers yourself
- Don't create bloated tasks
    - Limit of 32MB of task data per lease request
- Distribute task execution time, if possible.
- Choose a lease close to the worst-case scenario
- Catch transient and deadline exceeded exceptions and backoff

RESTful APIs for Pull Queue

Taskqueues

- get -- Get detailed information about a TaskQueue

Tasks

- delete -- Deletes a task from a TaskQueue
- get -- Gets the named task in a TaskQueue
- insert -- Insert a task into an existing queue
- lease -- Acquires a lease on the topmost N unowned tasks in a queue
- list -- Lists all non-deleted Tasks in a TaskQueue, whether or not they are currently leased, up to a maximum of 100
- patch -- Update tasks that are leased out of a TaskQueue
- update -- Update the duration of a task lease

Google App Engine

Cron And Scheduled Tasks

Google Cloud Platform

Cron Tasks

- Use cron for regularly scheduled tasks
- Cron tasks use a queue named "__cron"
- Cron job is executed as a GET request to the specified URL
- Configure in cron.xml or cron.yaml
  - URL
  - Schedule

Cron Configuration

- Url
  - The url to call (escape &, <, >, ', ")
- Schedule
  - The times/dates to execute the task
  - From/to or **synchronized** task
- Timezone
  - Optional, the standard zoneinfo name (defaults to UTC)
- Target
  - Optional, the target version of application (defaults to the default)

Cron Configuration Samples

Java cron.xml

```
<cronentries>
  <cron>
    <url>/recache</url>
    <description>
        Repopulate the cache
    </description>
    <schedule>
        every 2 minutes
    </schedule>
  </cron>
</cronentries>
```

Python cron.yaml

```
cron:
- description: new hourly summary
job
  url: /tasks/summary
  schedule: every 1 hours
```

! **url** and **schedule** are required parameters

Cron Tips

- Set security so only admins can access task handler URLs
- Go to the cron URL in a browser to test it
- When a cron task is executed, it has the header "X-AppEngine-Cron: true".
- You can have
  - 20 cron jobs for free apps
  - 100 for paid apps
- Review cron jobs in the Admin Console

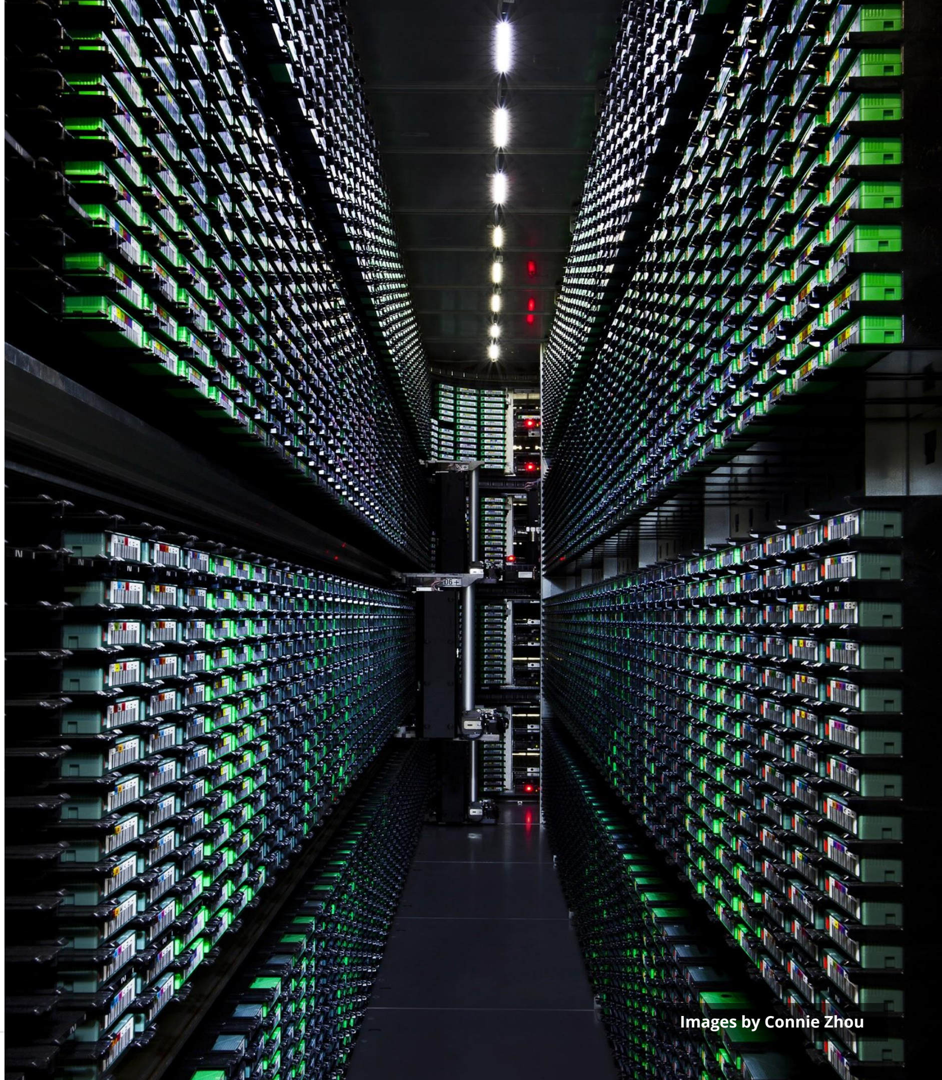
Quiz

If you wanted to share some work from App Engine with **Compute Engine** how could go about it using Task Queues?
(pick ***one*** *answer)*

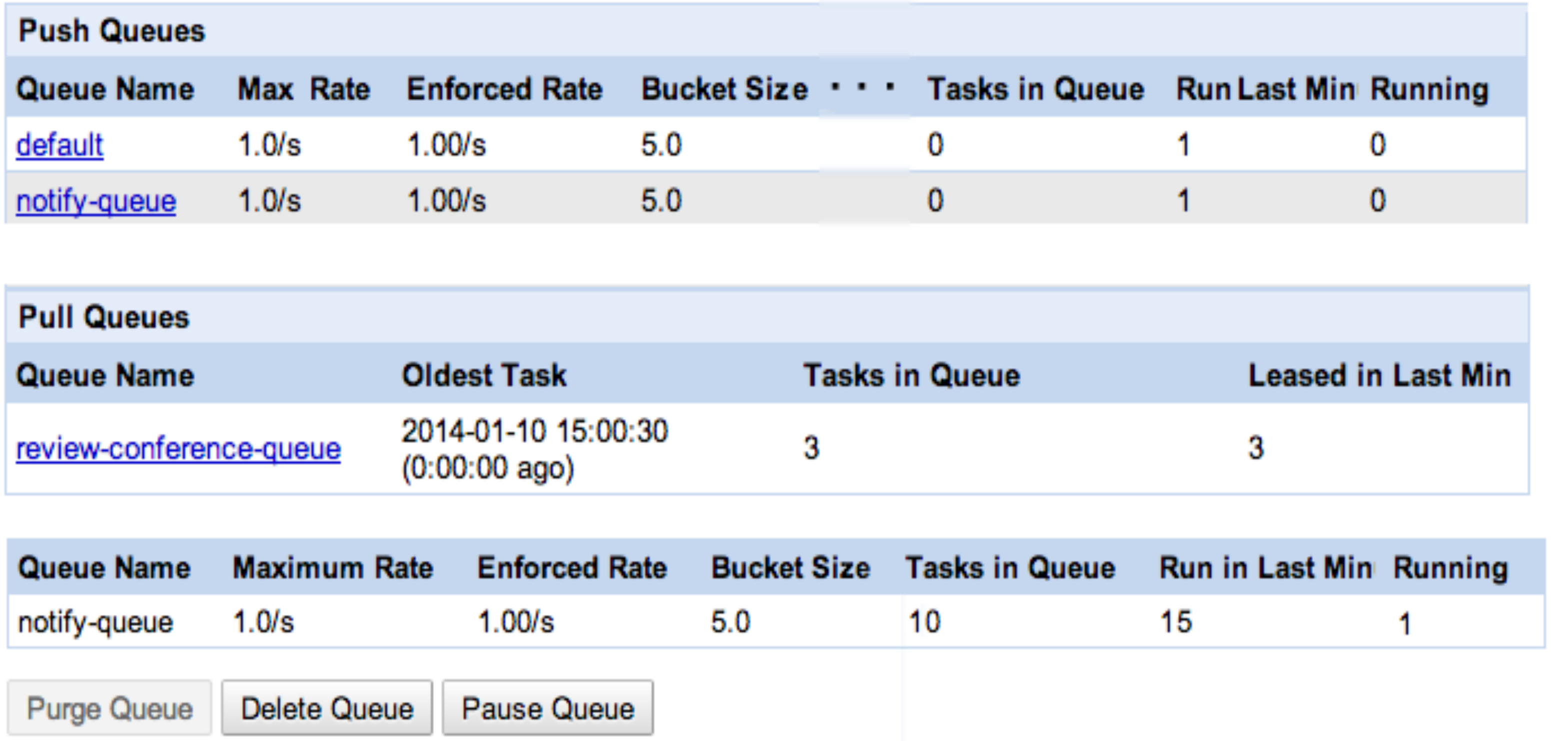
- ❏ A push queue
- ❏ Use Cron
- ❏ A scheduled task
- ❏ A pull queue
- ❏ All of the above
- ❏ It's not possible to share work in this way

Codelab

- Use a push queue to simulate sending out mail for a newly created conference
- When a conference is created, add a task to the task queue to notify all users about the new conference
- Write the handler to notify the appropriate users



Review Tasks in Admin Console



Push Queues

Queue Name	Max Rate	Enforced Rate	Bucket Size	. . .	Tasks in Queue	Run Last Min	Running
default	1.0/s	1.00/s	5.0		0	1	0
notify-queue	1.0/s	1.00/s	5.0		0	1	0

Pull Queues

Queue Name	Oldest Task	Tasks in Queue	Leased in Last Min
review-conference-queue	2014-01-10 15:00:30 (0:00:00 ago)	3	3

Queue Name	Maximum Rate	Enforced Rate	Bucket Size	Tasks in Queue	Run in Last Min	Running
notify-queue	1.0/s	1.00/s	5.0	10	15	1

Resources

- Configuring Queues (Java) (Python)
- Using Push Queue (Java) (Python)
- Using Pull Queues (Java) (Python)
- Article: Getting Started with Task Queues
- Backend Instances (Java) (Python)
- Scheduling tasks with cron (Java) (Python)

cloud.google.com

Images by Connie Zhou