**Проверка работоспособности**

Объясняет, как серверы gRPC предоставляют сервис проверки работоспособности и как клиент может быть настроен на автоматическую проверку работоспособности сервера, к которому он подключается.

Обзор

gRPC определяет стандартный сервис API (health/v1) для выполнения вызовов проверки работоспособности серверов gRPC. Реализация этого сервиса предоставляется, но вы несете ответственность за обновление состояния работоспособности ваших сервисов.

На стороне клиента вы можете настроить автоматическое взаимодействие клиента со службами мониторинга работоспособности ваших серверных приложений. Это позволит клиенту избегать служб, которые считаются неработоспособными.

Служба работоспособности на стороне сервера

Служба проверки работоспособности на сервере gRPC поддерживает два режима работы:

* Унарные вызовы к Check конечной точке rpc
  + Полезно для централизованного мониторинга или балансировки нагрузки, но не подходит для поддержки множества клиентов gRPC, постоянно выполняющих проверку работоспособности
* Потоковая передача обновлений работоспособности с помощью Watch конечной точки rpc
  + Используется функцией проверки работоспособности на стороне клиента в клиентах gRPC

Включение службы проверки работоспособности на вашем сервере включает в себя следующие шаги:

1. Используйте предоставленную библиотеку проверки работоспособности для создания службы проверки работоспособности
2. Добавьте службу проверки работоспособности на свой сервер.
3. Уведомляйте библиотеку проверки работоспособности при изменении работоспособности одной из ваших служб.
   * NOT\_SERVING если ваш сервис в данный момент не может принимать запросы
   * SERVING если ваш сервис открыт для бизнеса
   * Если вас не волнует работоспособность отдельных служб, вы можете использовать пустую строку (""), чтобы обозначить работоспособность всего сервера.
4. Обязательно сообщите библиотеке проверки работоспособности о завершении работы сервера, чтобы она могла уведомить всех подключённых клиентов.

Точные сведения зависят от языка, см. раздел **«Поддержка языков»** ниже.

Включение проверки работоспособности клиента

Клиент gRPC можно настроить для выполнения проверок работоспособности серверов, к которым он подключается, изменив конфигурацию сервиса канала. Например, для мониторинга работоспособности сервиса foo можно использовать (в формате JSON):

{

**"healthCheckConfig"**: {

**"serviceName"**: "foo"

}

}

Обратите внимание, что если ваш сервер сообщает о работоспособности службы пустой строки (""), что означает работоспособность всего сервера, вы также можете использовать пустую строку здесь.

Enabling health checking changes some behavior around calling a server:

* The client will additionally call the Watch RPC on the health check service when a connection is established
  + If the call fails, retries will be made (with exponential backoff), unless the call fails with the status UNIMPLEMENTED, in which case health checking will be disabled.
* Requests won’t be sent until the health check service sends a healthy status for the service being called
* If a healthy service becomes unhealthy the client will no longer send requests for that service
* The calls will resume if the service later becomes healthy
* Some load balancing policies can choose to disable health checking if the feature does not make sense with the policy (e.g. pick\_first does this)

Если говорить более конкретно, то состояние субканала (который представляет собой физическое соединение с сервером) переходит из одного состояния в другое в зависимости от работоспособности службы, к которой он подключается.

Connection requestedHealth check:  
SERVINGHealth check:  
NOT\_SERVING  
or call failsHealth check:  
NOT\_SERVINGConnection breaks  
or times outHealth check:  
SERVINGIDLECONNECTINGREADYTRANSIENT\_FAILUREAllows the load balancer to choose  
another, working subchannel

Опять же, особенности включения проверки работоспособности на стороне клиента зависят от языка. Примеры см. в разделе **Поддержка языков**.

Языковая Поддержка

ЯзыкПримерJavaПример JavaВпередПерейти к примеруПитонПример PythonC++Пример C ++

Последнее изменение 20 мая 2024 г.: добавлена ссылка на пример проверки работоспособности C++ (#1289) (571730d)