大数点Cloud API使用手册

Contents

[Revision History 3](#_Toc435602795)

[简介 3](#_Toc435602796)

[开发语言要点 4](#_Toc435602797)

[Dasudian Cloud API使用规范参考 4](#_Toc435602798)

[服务初始化（initialization） 4](#_Toc435602799)

[服务结束（termination） 5](#_Toc435602800)

[订阅（subscribe） 5](#_Toc435602801)

[订阅次数（subscribe\_count） 7](#_Toc435602802)

[订阅取消（unsubscribe） 7](#_Toc435602803)

[同步请求发送(send\_sync) 8](#_Toc435602804)

[异步请求发送（send\_aysnc） 9](#_Toc435602805)

[服务请求发布（publish） 10](#_Toc435602806)

[异步接收返回（recv\_async） 11](#_Toc435602807)

[同步返回（return） 12](#_Toc435602808)

[转发（forward） 13](#_Toc435602809)

[Dasudian Cloud数据库使用规范 14](#_Toc435602810)

[DSD NoSQL 14](#_Toc435602811)

[Mysql 15](#_Toc435602812)

# Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| Revision | Date | Author | Remark |
| PA1 | 2015-09-26 | Barco You | Cloud API functions are described |
| PA2 | 2015-11-12 | Barco You | Cloud DBs are added |

# 简介

Dasudian Cloud API是一种基于简单消息模型的服务API，帮助服务构建者快速、简单地实现各种业务逻辑的REST API服务和Real-Time API服务。大数点Cloud API同时支持Publish/Subscribe和Request/Response两种通信模型，并且可基于我们提供的SDK使用任何一种支持的开发语言实现，目前包括：Java、C/C++、PHP、Javascript、Python、Ruby、Go和Perl。通过任何一种支持语言实现的API，经过编译后都可以运行在Dasudian API容器（DAC）内。而Dasudian API容器为大数点IoT云平台的一个基础PaaS服务，基于电信级服务开发平台Erlang实现，将合作伙伴实现的Cloud API运行在大数点分布在世界各地的计算容器内，帮助合作伙伴快速实现业务扩展、业务定点部署和业务升级。

Dasudian Cloud API包含以下特性，帮助您快速实现业务开发、部署和实现企业级SOA（Service Oriented Architecture）：

* 每个服务API只需要实现最少的Callback函数（仅一个请求回调函数）以响应服务请求，保持通过Dasudian Cloud API实现的服务简单，所以相对于其他解决方案错误几率更低；
* 每个服务API可以独立指定权重、服务目标等，实现服务能力的高粒度优化；
* 每个服务可根据接收到的请求决定是否返回（Response）；
* 每个请求并不需持久化到数据库，以防由其在不同环境下引起的不可预知且不易察觉的错误；
* 通过各种语言实现的服务API都由Erlang进程统一调度，保证一致的性能与容错能力；
* 每个服务API可以指定一个优先级，为业务响应提供灵活配置；
* 每个服务请求都包含一个timeout时间，可以根据业务处理逻辑调节、优化响应的允许延迟时间。

大数点Cloud API使用统一的通信原语subscribe、send\_sync和send\_aysnc(在不同的开发语言里有对应的函数)分别实现Pub/Sub的消息传输和点对点的消息传输，其中send\_sync和send\_async分别实现有响应的同步消息发送与异步消息发送。这三个函数都是对某个服务名称（API名称，如：“get\_user\_list”）表达式进行作用，当一个服务请求发送到对应的服务/API名称时，该请求就会被subscribe、send\_sync或send\_async了该服务名称的**服务进程**（在Dasudian API容器中运行的Cloud API实例）所接收到并进行处理。subscribe的服务名称表达式可以为 ”\*”，表示匹配所有服务请求，若多个API服务进程subscribe了同一个服务名称，则对应的请求被随机分配到一个服务进程处理。

通信原语publish、return、和recv\_async分别对应subscribe、send\_sync和send\_aysnc。publish用于向subscribe了某服务名称的服务进程或外部应用发布消息； return用于响应一个服务请求并返回数据，return可以返回一个null response如果服务请求者并不需要一个返回，调用return后该服务进程结束；recv\_async接收来自于send\_async发出的异步服务请求或消息。forward函数将服务请求转发给下一个服务进行处理，并结束自己的服务进程。

# 开发语言要点

C/C++基于同一个SDK包，底层由一套代码实现，只是将C的指针封装成了C++ API对象，所以基于二者开发的服务在性能上没有区别。为了避免使用libstdc++和内部的内存池，应避免适应STL。C++ API的命名空间为dsd\_cloud，且定义STRING数据类型对应char const \* const 或者std::string const &，二者在重载函数里都被支持。

Java SDK没有集成C/C++的代码，其通过反射来使用底层的文件描述对象和存储对象函数指针，实现服务时通过引入com.dasudian.API来引入相关的API函数。

Python SDK提供dsd\_cloud module和dsd\_cloud\_c module，其中dsd\_cloud\_c module基于C++的API实现，效率相对于纯Python的dsd\_cloud会高一些。

# Dasudian Cloud API使用规范参考

使用大数点Cloud API，首先需要到大数点开发者平台(<https://dev.dasudian.com>)注册成为大数点云平台用户并登录，然后创建相应的API。开发者可以为每个API指定其名称、协议（HTTP/HTTPS/CoAP/Websocket等），访问路径、内存分配、最大线程数、请求超时、优先级等，并选择API的运行时环境（开发语言），具体请访问大数点官网文档（<https://www.dasudian.com/doc>）。

## 服务初始化（initialization）

凡是在大数点API容器内运行的API，无论是通过何种语言实现，都可以在API容器内指定给API分配的内存空间和线程数(决定该API的最大并发数和响应延时)，该配置可以通过登录大数点开发者门户([https://dev.dasudian.com](https://join.dasudian.com))实现。在大数点提供的Cloud API开发SDK里，也包含了对应的函数，用户获取服务配置的线程数，分别如下：

|  |  |
| --- | --- |
| **开发语言** | **API函数** |
| C | int dsd\_cloud\_initialize\_thread\_count(unsigned int \* const thread\_count); |
| C++ | unsigned int dsd\_cloud::API::thread\_count(); |
| JAVA | int com.dasudian.API.thread\_count(); |
| Python | dsd\_cloud.API.thread\_count()  dsd\_cloud\_c.API.thread\_count() |
| Ruby | dsd\_cloud::API.thread\_count() |

每个API服务还可以配置一个timeout（毫秒）时间来限制其在API容器内创建（初始化）的时间，一个API服务初始化后，首先执行的是poll()函数。即timeout限制的是服务初始化到执行poll函数的时间，用于接受对服务的请求并等待服务初始完成，直至设置的timeout超时，或服务终止；如果timeout设置为0，则poll函数立即返回；如果timeout没有在大数点开发者门户(<https://join.dasudian.com>)进行配置，则默认值为-1，表示poll函数将无限阻塞终止服务终止termimation()。poll()函数的返回值为boolean或int类型（C/C++, JAVA）,返回值true（或非0）表示发生超时，false（或0）标识该服务执行至结束。

在服务初始化过程中，以下API函数不可以被执行：send\_sync，recv\_async，return和forward，其中，send\_sync和recv\_async分别阻塞执行等待接收服务响应和请求，而服务初始化必须是非阻塞的且在timeout时间内完成；return和forward函数只有在完成对服务请求的处理后才能有效执行。

如果初始化超时，则该服务失败且可以被重启。在服务配置里，MaxR设定该服务可以被重启的最大次数，MaxT（秒）指定重启的时限，及在MaxT秒内最多可以被重启MaxR次。如果该服务是配置在API容器的全局配置文件（e.g. /usr/local/etc/dac/dac.conf）里，当该服务失败并试图重启MaxR次时，整个API容器则自动停止，以防止其他错操作。

每个API服务都可以配置一个ACL（Access Control List）来定义服务的名称规范，来允许或拒绝服务请求。

## 服务结束（termination）

Termination函数用于服务清理相关状态（比如断开连接或关掉文件等），在termination函数内，不可以执行SDK提供的任何Cloud API函数。

## 订阅（subscribe）

对某个服务名称表达式进行订阅，并对相应的服务请求（只要请求的服务名称与订阅的名称表达式相匹配）进行处理和响应。服务名称表达式为一个字符串，可以包含“\*”，表示匹配1个或多个任何字符；但是，“\*\*”为非法表达式。服务名称表达式一般为一个文件路径形式（如：”/dasudian/app\_server/login”），为了更好地设计您的API服务，服务名称表达式应该反映服务提供的域或范围，并有相应层次结构，可参考Uniform Resource Identifier (URI)进行相关规范设计。

当一个服务订阅了某个服务名称表达式，该服务名称表达式将被拼接在一个“前缀“后面形成一个完成的服务URL的PATH部分，该服务容器由大数点开发者平台根据开发者信息和所创建的应用信息自动生成，唯一标识一个开发者的一个应用，如：”/dsd2345/authentication/login”中的前缀”dsd2345”。

同一个服务名称表达式可以被多个服务同时sbscribe，但在一个服务里包含多个subscribe语句可以增加该服务接收到对应服务请求的概率。

各开发语言对应的API函数为：

|  |  |
| --- | --- |
| **开发语言** | **API函数** |
| C | int dsd\_cloud\_subscribe(dsd\_cloud\_instance\_t \* p,  char const \* const pattern,  dsd\_cloud\_callback\_t f); |
| C++ | template <typename T>  int dsd\_cloud::API::subscribe(STRING pattern,  T & object,  void (T::\*f) (dsd\_cloud::API const &,  int const,  STRING,  STRING,  void const \* const,  uint32\_t const,  void const \* const,  uint32\_t const,  uint32\_t,  int8\_t,  char const \* const,  char const \* const,  uint32\_t const)) const;  int dsd\_cloud::API::subscribe(STRING pattern,  void (\*f) (API const &,  int const,  STRING,  STRING,  void const \* const,  uint32\_t const,  void const \* const,  uint32\_t const,  uint32\_t,  int8\_t,  char const \* const,  char const \* const,  uint32\_t const)) const |
| JAVA | void com.dasudian.API.subscribe(final String pattern,  final Object instance,  final String methodName); |
| Python | dsd\_cloud\_c.API.subscribe(pattern, Function)  dsd\_cloud.API.subscribe(pattern, Function) |
| Ruby | dsd\_cloud::API.subscribe(pattern, function) |

## 订阅次数（subscribe\_count）

该函数返回一个计数，标明某个服务名称被当前服务进程订阅的次数。一般该返回值为0或1，但是在同一服务内，某服务名称可以被订阅多次，以提高对应的服务请求被该服务进程接收到的几率，如果该服务名称被多个服务订阅的话。subscribe\_count在测试或程序调试时非常有用，但在线上服务也可以被使用，如果服务中subscribe模式非常复杂，该函数可用于跟踪每个subscribe操作。

各开发语言对应的API函数为：

|  |  |
| --- | --- |
| **开发语言** | **API函数** |
| C | //订阅某服务  int dsd\_cloud\_subscribe\_count(dsd\_cloud\_instance\_t \* p,  char const \* const pattern);  //获取订阅次数  dsd\_cloud\_get\_subscribe\_count(p) |
| C++ | //订阅某服务  int dsd\_cloud::API::subscribe\_count(STRING pattern) const;  //获取订阅次数  dsd\_cloud::API::get\_subscribe\_count() to get the result |
| JAVA | //获取订阅次数  int com.dasudian.API.subscribe\_count(final String pattern);  // 返回值即为订阅次数 |
| Python | dsd\_cloud\_c.API.subscribe\_count(pattern)  dsd\_cloud.API.subscribe\_count(pattern)  # 返回值为订阅次数 |
| Ruby | dsd\_cloud::API.subscribe\_count(pattern)  #返回值为订阅次数 |

## 订阅取消（unsubscribe）

Unsubscibe将取消对某个服务名称的订阅，若该服务多次订阅了某个服务名称，则该unsubscribe操作仅取消其中的一个订阅实例，而被取消的订阅实例对应取消后第一次被服务请求匹配到的subscribe操作。

各开发语言对应的API函数为：

|  |  |
| --- | --- |
| **开发语言** | **API函数** |
| C | int dsd\_cloud\_unsubscribe(dsd\_cloud\_instance\_t \* p,  char const \* const pattern); |
| C++ | int dsd\_cloud::API::unsubscribe(STRING pattern) const; |
| JAVA | void com.dasudian.API.unsubscribe(final String pattern); |
| Python | dsd\_cloud\_c.API.unsubscribe(pattern)  dsd\_cloud.API.unsubscribe(pattern) |
| Ruby | dsd\_cloud::API.unsubscribe(pattern) |

## 同步请求发送(send\_sync)

向某个服务名称发送一个同步（阻塞式）请求，且可以设置超时和优先级，若不指定timeout，则给该服务配置的超时时间（在大数点开发者平台https://dev.dasudian.com上对某个服务API的配置）被使用。优先级也可以在大数点开发这平台进行配置，默认值为0（标识最高优先级，254表示最低优先级）。

各开发语言对应的API函数为：

|  |  |  |
| --- | --- | --- |
| **语言** | **API函数** | **返回值** |
| C | int dsd\_cloud\_send\_sync\_(dsd\_cloud\_instance\_t \* p,  char const \* const name,  void const \* const request\_info,  uint32\_t const request\_info\_size,  void const \* const request,  uint32\_t const request\_size,  uint32\_t timeout,  int8\_t const priority); | 0-发送成功  非0-发送失败 |
| C++ | int dsd\_cloud::API::send\_sync(STRING name,  void const \* const request\_info,  uint32\_t const request\_info\_size,  void const \* const request,  uint32\_t const request\_size,  uint32\_t timeout,  int8\_t const priority) const; | 0-发送成功  非0-发送失败 |
| JAVA | Response com.dasudian.API.send\_sync(String name, byte[] request\_info, byte[] request, Integer timeout, Byte priority); | 响应数据为该函数的返回值:com.dasudian.API.Response对象 |
| Python | dsd\_cloud\_c.API.send\_sync(name, request, timeout=None, request\_info=None, priority=None)  dsd\_cloud.API.send\_sync(name, request, timeout=None, request\_info=None, priority=None) | 响应数据为该函数的返回值: (response\_info, response, trans\_id) |
| Ruby | dsd\_cloud::API.send\_sync(name, request, timeout=nil, request\_info=nil, priority=nil) | 响应数据为该函数的返回值: [response\_info, response, trans\_id] |

获取send\_sync发送成功后响应的数据：

|  |  |
| --- | --- |
| **语言** | **API函数** |
| C | dsd\_cloud\_get\_response(p)  dsd\_cloud\_get\_response\_size(p)  dsd\_cloud\_get\_response\_info(p)  dsd\_cloud\_get\_response\_info\_size(p)  dsd\_cloud\_get\_trans\_id\_count(p)  dsd\_cloud\_get\_trans\_id(p, i) |
| C++ | char const \* dsd\_cloud::API::get\_response() const;  uint32\_t dsd\_cloud::API::get\_response\_size() const;  char const \* dsd\_cloud::API::get\_response\_info() const;  uint32\_t dsd\_cloud::API::get\_response\_info\_size() const;  uint32\_t dsd\_cloud::API::get\_trans\_id\_count() const;  char const \* dsd\_cloud::API::get\_trans\_id(unsigned int const i = 0) const; |
| JAVA | dsd.dasudian.API.Response 对象 |
| Python | (response\_info, response, trans\_id) |
| Ruby | [response\_info, response, trans\_id] |

无论何种开发语言，一组send\_sync发送和返回由一个唯一的trans\_id标识，服务可基于该标识对响应数据进行处理和存储。

## 异步请求发送（send\_aysnc）

该函数对某个服务名称发送一个异步（非阻塞）请求，同send\_sync一样，也可以指定timeout和优先级，且都可以在大数点开发者平台上对相应的服务的timeout和优先级进行配置，默认优先级为0（最高优先级，最低优先级为254）。一旦一步请求被发送，send\_aysnc立即返回一个transId，唯一标识此次请求并用于获取对应得返回数据。

各开发语言对应的API函数为：

|  |  |  |
| --- | --- | --- |
| **语言** | **API函数** | **返回值** |
| C | int dsd\_cloud\_send\_async\_(dsd\_cloud\_instance\_t \* p,  char const \* const name,  void const \* const request\_info,  uint32\_t const request\_info\_size,  void const \* const request,  uint32\_t const request\_size,  uint32\_t timeout,  int8\_t const priority); | 0-发送成功  非0-发送失败 |
| C++ | int dsd\_cloud::API::send\_async(STRING name,  void const \* const request\_info,  uint32\_t const request\_info\_size,  void const \* const request,  uint32\_t const request\_size,  uint32\_t timeout,  int8\_t const priority) const; | 0-发送成功  非0-发送失败 |
| JAVA | TransId com.dasudian.API.send\_async(String name, byte[] request\_info, byte[] request, Integer timeout, Byte priority); | com.dasudain.API.TransId 对象 |
| Python | dsd\_cloud\_c.API.send\_async(name, request, timeout=None, request\_info=None, priority=None)  dsd\_cloud.API.send\_async(name, request, timeout=None, request\_info=None, priority=None) | trans\_id |
| Ruby | dsd\_cloud::API.send\_async(name, request,  timeout=nil, request\_info=nil, priority=nil) | trans\_id |

获取send\_async发送成功后对应响应的数据：

|  |  |
| --- | --- |
| **语言** | **API函数** |
| C | dsd\_cloud\_get\_trans\_id\_count(p)  dsd\_cloud\_get\_trans\_id(p, i) |
| C++ | uint32\_t dsd\_cloud::API::get\_trans\_id\_count() const;  char const \* dsd\_cloud::API::get\_trans\_id(unsigned int const i = 0) const; |
| JAVA | com.dasudian.API.TransId 对象 |
| Python | trans\_id |
| Ruby | trans\_id |

## 服务请求发布（publish）

消息发布为pub/sub模式中对应于subscribe的另一端，其实一种异步式消息组播，但可以用于对服务请求进行相应。该函数主要用于向所有订阅（subscribe）了某个或一些服务的目标发送一个异步的服务请求。Publish函数的返回值为一组TransId，分别标识由publish操作发出的每一个服务请求。对于接收publish服务请求的服务应该返回一个null响应，以表示对应的服务请求并不期望获得任何响应。

各开发语言对应的API函数为：

|  |  |  |
| --- | --- | --- |
| **语言** | **API函数** | **返回值** |
| C | int dsd\_cloud\_publish\_(dsd\_cloud\_instance\_t \* p,  char const \* const name,  void const \* const request\_info,  uint32\_t const request\_info\_size,  void const \* const request,  uint32\_t const request\_size,  uint32\_t timeout,  int8\_t const priority); | 0-发送成功  非0-发送失败 |
| C++ | int dsd\_cloud::API::publish(STRING name,  void const \* const request\_info,  uint32\_t const request\_info\_size,  void const \* const request,  uint32\_t const request\_size,  uint32\_t timeout,  int8\_t const priority) const; | 0-发送成功  非0-发送失败 |
| JAVA | List<TransId> com.dasudian.API.mcast\_async(String name, byte[] request\_info, byte[] request, Integer timeout, Byte priority); | com.dasudain.API.TransId 对象List |
| Python | dsd\_cloud\_c.API.mcast\_async(name, request, timeout=None, request\_info=None, priority=None)  dsd\_cloud.API.mcast\_async(name, request, timeout=None, request\_info=None, priority=None) | [trans\_id] |
| Ruby | dsd\_cloud::API.mcast\_async(name, request,  timeout=nil, request\_info=nil, priority=nil) | [trans\_id] |

获取publish发送成功后对应响应的数据：

|  |  |
| --- | --- |
| **语言** | **API函数** |
| C | dsd\_cloud\_get\_trans\_id\_count(p)  dsd\_cloud\_get\_trans\_id(p, i) |
| C++ | uint32\_t dsd\_cloud::API::get\_trans\_id\_count() const;  char const \* dsd\_cloud::API::get\_trans\_id(unsigned int const i = 0) const; |
| JAVA | List<com.dasudian.API.TransId> |
| Python | [trans\_id] |
| Ruby | [trans\_id] |

## 异步接收返回（recv\_async）

该函数用于接收指定的异步服务请求send\_async的响应数据（由TransId标识），若未指定TransId，一个null ID用于获取最早的一个异步请求响应，除非该响应已经超时。默认情况下，recv\_async具有排他性，被其获取过数据的服务响应不能再次获取。recv\_async函数仍返回TransId，唯一标识一组请求和响应。

各开发语言对应的API函数为：

|  |  |  |
| --- | --- | --- |
| **语言** | **API函数** | **返回值** |
| C | int dsd\_cloud\_recv\_async(dsd\_cloud\_instance\_t \* p,  uint32\_t timeout,  char const \* const trans\_id,  int consume); | 0-发送成功  非0-发送失败 |
| C++ | int dsd\_cloud::API::recv\_async(uint32\_t timeout,  STRING trans\_id,  bool consume) const; | 0-发送成功  非0-发送失败 |
| JAVA | Response com.dasudian.API.recv\_async(Integer timeout, byte[] transId, boolean consume); | com.dasudain.API.Response 对象 |
| Python | dsd\_cloud\_c.API.recv\_async(timeout=None, trans\_id=None, consume=True)  dsd\_cloud.API.recv\_async(timeout=None, trans\_id=None, consume=True) | (response\_info, response, trans\_id) |
| Ruby | dsd\_cloud::API.recv\_async(timeout=nil, trans\_id=nil, consume=true) | [response\_info, response, trans\_id] |

获取publish发送成功后对应响应的数据：

|  |  |
| --- | --- |
| **语言** | **API函数** |
| C | dsd\_cloud\_get\_response(p)  dsd\_cloud\_get\_response\_size(p)  dsd\_cloud\_get\_response\_info(p)  dsd\_cloud\_get\_response\_info\_size(p)  dsd\_cloud\_get\_trans\_id\_count(p)  dsd\_cloud\_get\_trans\_id(p, i) |
| C++ | char const \* dsd\_cloud::API::get\_response() const;  uint32\_t dsd\_cloud::API::get\_response\_size() const;  char const \* dsd\_cloud::API::get\_response\_info() const;  uint32\_t dsd\_cloud::API::get\_response\_info\_size() const;  uint32\_t dsd\_cloud::API::get\_trans\_id\_count() const;  char const \* dsd\_cloud::API::get\_trans\_id(unsigned int const i = 0) const;  char const \* dsd\_cloud::API::get\_trans\_id(unsigned int const i = 0) const; |
| JAVA | com.dasudian.API.Response |
| Python | (response\_info, response, trans\_id) |
| Ruby | [response\_info, response, trans\_id] |

## 同步返回（return）

对一个同步服务请求返回数据，在return返回数据后将抛出一个异常（exception）用于通知服务请求的处理者（handler）结束进程。在对于Java、Python和Ruby语言实现的API服务，可以直接将服务请求handler的返回作为对服务请求响应数据，而最直观的返回方式就是调用return语句。

各开发语言对应的API函数为：

|  |  |
| --- | --- |
| **语言** | **API函数** |
| C | int dsd\_cloud\_return(dsd\_cloud\_instance\_t \* p,  int const command,  char const \* const name,  char const \* const pattern,  void const \* const response\_info,  uint32\_t const response\_info\_size,  void const \* const response,  uint32\_t const response\_size,  uint32\_t timeout,  char const \* const trans\_id,  char const \* const pid,  uint32\_t const pid\_size); |
| C++ | int dsd\_cloud::API::return\_(int const command,  STRING name,  STRING pattern,  void const \* const response\_info,  uint32\_t const response\_info\_size,  void const \* const response,  uint32\_t const response\_size,  uint32\_t timeout,  char const \* const trans\_id,  char const \* const pid,  uint32\_t const pid\_size) const |
| JAVA | void com.dasudian.API.return\_(Integer command,  String name, String pattern,  byte[] response\_info, byte[] response,  Integer timeout, byte[] transId,  OtpErlangPid pid); |
| Python | dsd\_cloud\_c.API.return\_(command, name, pattern, response\_info, response,  timeout, trans\_id, pid)  dsd\_cloud.API.return\_(command, name, pattern, response\_info, response,  timeout, trans\_id, pid) |
| Ruby | dsd\_cloud::API.return\_(command, name, pattern, response\_info, response,  timeout, trans\_id, pid) |

## 转发（forward）

将服务请求转发到其他服务地址，且forward可以改变原服务请求的相关参数，形成一个新的不同的请求。forward函数转发请求后会抛出一个异常（exception）用以结束掉当前服务进程。

各开发语言对应的API函数为：

|  |  |
| --- | --- |
| **语言** | **API函数** |
| C | int dsd\_cloud\_forward(dsd\_cloud\_instance\_t \* p,  int const command,  char const \* const name,  void const \* const request\_info,  uint32\_t const request\_info\_size,  void const \* const request,  uint32\_t const request\_size,  uint32\_t timeout,  int8\_t const priority,  char const \* const trans\_id,  char const \* const pid,  uint32\_t const pid\_size); |
| C++ | int dsd\_cloud::API::forward\_(int const command,  STRING name,  void const \* const request\_info,  uint32\_t const request\_info\_size,  void const \* const request,  uint32\_t const request\_size,  uint32\_t timeout,  int8\_t const priority,  char const \* const trans\_id,  char const \* const pid,  uint32\_t const pid\_size) const; |
| JAVA | Response com.dasudian.API.forward\_(Integer command, String name,  byte[] request\_info, byte[] request,  Integer timeout, Byte priority,  byte[] transId, OtpErlangPid pid); |
| Python | dsd\_cloud\_c.API.forward\_(command, name, request\_info, request,  timeout, priority, trans\_id, pid)  dsd\_cloud.API.forward\_(command, name, request\_info, request,  timeout, priority, trans\_id, pid) |
| Ruby | dsd\_cloud::API.forward\_(command, name, request\_info, request,  timeout, priority, trans\_id, pid) |

# Dasudian Cloud数据库使用规范

使用大数点Cloud DB，首先需要到大数点开发者平台(<https://dev.dasudian.com>)注册成为大数点云平台用户并登录，然后创建相应的数据库服务。开发者可以为每数据库服务指定其数据库类型（DSD NoSQL、Mysql、Postgresql、Cassandra、MongoDB等），并配置数据库的名称、存储容量、最大访问并发数等，具体请访问大数点官网文档（<https://www.dasudian.com/doc>）。

所有Cloud DB都以服务的形式提供给开发者，

## DSD NoSQL

DSD NoSQL为大数点自行研发的高性能分布式NoSQL数据库

## Mysql