# 镜像启动命令：

docker run -itd -p 8083:8083 image\_server:4.0.1 ./server

镜像内部默认8083端口，-v 需要添加数据目录映射

可选默认参数：

--port 8083

--thread 12

--statistic\_size 1024

--gdal\_cache\_size “1000”

--use\_etcd\_v2 或者 --use\_etcd\_v3 注：v3只能在linux下用

--etcd\_v2\_host “127.0.0.1”

--etcd\_v2\_port “2379”

--etcd\_v3\_address “http://127.0.0.1:4001” 注：如果有多个ip，用逗号隔开

如果使用集群部署方式，即开启了多个影像服务，并且设定了"style"，则需要开启etcd。因为渲染"style"需要保存在etcd中。如果只开启一个服务，但是设定了"style"，也是不需要开启etcd的。

## 不使用etcd举例：

docker run -itd -p 8083:8083 –v /mnt:/mnt image-server-gdal3-etcd3-aws:1.0.5 ./server

## 使用etcd举例：

启动etcd：

docker run --rm -d --network app-tier --name etcd-server --publish 2379:2379 --publish 2380:2380 --env ALLOW\_NONE\_AUTHENTICATION=yes --env ETCD\_ADVERTISE\_CLIENT\_URLS=http://etcd-server:2379 bitnami/etcd:latest

启动服务：

docker run -d -p8083:8083 --rm --network app-tier -v/mnt:/mnt image-server-gdal3-etcd3-aws:1.0.5 ./server --use\_etcd\_v3 --etcd\_v3\_address <http://etcd-server:2379>

## 支持aws s3存储：

### 不使用缓存

docker run -d --rm -p30803:8083 -v/data:/data -e AWS\_REGION=cn-northwest-1 -e AWS\_SECRET\_ACCESS\_KEY=<key> -e AWS\_ACCESS\_KEY\_ID=<key> -e AWS\_S3\_ENDPOINT=s3.cn-northwest-1.amazonaws.com.cn image-server-gdal3-etcd3-aws:1.0.5 ./server

### 使用缓存

可以指定一个bucket用来写入缓存，把请求的瓦片返回的同时，写入s3，下次请求的时候，会直接到s3去找该瓦片

启动：docker run -it --rm -p30014:8083 -v/mnt/data:/mnt/data -e AWS\_REGION=cn-northwest-1 -e AWS\_SECRET\_ACCESS\_KEY=uGXq6F4CXnVsRXTU/bLiBFJLjgpD+MPFrTM+z13e -e AWS\_ACCESS\_KEY\_ID=AKIAT2NCQYSI3X7D52BZ -e AWS\_S3\_ENDPOINT=s3.cn-northwest-1.amazonaws.com.cn image-server-gdal3-etcd3-aws:1.0.5 ./server --use\_amazon\_s3 --amazon\_s3\_bucket\_name "pie-engine-test/NN/image-server-cache1"

前端需要加参数cache=1，如果不加，则不使用缓存：http://161.189.202.63:30014/v1?request=GetTile&TILECOL={x}&TILEROW={y}&TILEMATRIX={z}&cache=1&info={"info":[{"path":"/vsis3/pie-engine-test/NN/world.tif"},{"path":"/vsis3/pie-engine-test/NN/mosaic.tif"}]}

瓦片会写入到s3 的pie-engine-test/NN/image-server-cache1下面。瓦片是可以复用的，使用md5加密命名，根据"info"内容和索引号生成的md5命名瓦片

## 使用aws s3自定义瓦片缓存

如果使用了aws-tile工具创建了缓存，例如

docker run -itd --rm aws-tile:1.0.0 ./aws-tile "/vsis3/pie-engine-test/NN/DEM-Gloable32.tif" "4" "4" "cn-northwest-1" "uGXq6F4CXnVsRXTU/bLiBFJLjgpD+MPFrTM+z13e" "AKIAT2NCQYSI3X7D52BZ" "s3.cn-northwest-1.amazonaws.com.cn" "pie-engine-test/make-tile-test" "780" "0"

则参数 "pie-engine-test/make-tile-test" 需要 在下面前端的地址的"s3cachekey"里写上对应的值。如果前端url不写"s3cachekey"或者"s3cachekey":""，则认为不使用瓦片缓存

[http://161.189.202.63:30010/v1?request=GetTile&TILECOL={x}&TILEROW={y}&TILEMATRIX={z}&info={"info":[{"path":"/vsis3/pie-engine-test/NN/DEM-Gloable32.tif","s3cachekey":"pie-engine-test/make-tile-test"}]}](http://161.189.202.63:30010/v1?request=GetTile&TILECOL=%7bx%7d&TILEROW=%7by%7d&TILEMATRIX=%7bz%7d&info=%7b%22info%22:%5b%7b%22path%22:%22/vsis3/pie-engine-test/NN/DEM-Gloable32.tif%22,%22s3cachekey%22:%22pie-engine-test/make-tile-test%22%7d%5d%7d)

该算法已经封装为aws lambda上传到了aws。1.3节的缓存是动态生成瓦片的同时写入s3，提升下次访问的速度，并不能提升首次瓦片生成的响应速度。aws-tile是预先生成原始数据的切片，可以提升首次及后续访问响应速度。

# WMTS前端调用两种方式：

## 前端调用方式一：每次请求时指定请求的数据路径和style(可以省略)，有GET和POST两种方法。支持集群部署，不需要etcd。"info"字段为json数组，支持单张或多张影像。GET和POST内容是一致的，均为json格式。

### GET

http://localhost:8083/v1?request=GetTile&TILECOL={x}&TILEROW={y}&TILEMATRIX={z}&info={info":[{"path":"d:/linux\_share/DEM-Gloable32.tif","style":{"stretch":{"kind":"histogramEqualize","percent":0.0}}},{"path":"d:/linux\_share/t/1.tiff","style":{"stretch":{"kind":"percentMinimumMaximum","percent":3.0}}}],"format":"png"}

format支持"webp"和"png"两种，默认是"webp"。style和format可以省略：

http://localhost:8083/v1?request=GetTile&TILECOL={x}&TILEROW={y}&TILEMATRIX={z}&info={"info":[{"path":"d:/linux\_share/DEM-Gloable32.tif"},{"path": "d:/linux\_share/t/1.tiff"}]}

设置影像外部无效值，去黑边

http://10.1.33.45:8083/v1?cache=false&request=GetTile&TILECOL={x}&TILEROW={y}&TILEMATRIX={z}&info={"info":[{"path":"/mnt/hgfs/Z/ai\_data/source\_data/20210314/123/北京市.img","style":{"stretch":{"externalNodataValue":0}}}]}

举例: mapbox前端代码：



### POST

http://localhost:8083/v1?request=GetTile&TILECOL={x}&TILEROW={y}&TILEMATRIX={z}

postbody体内容：{"info":[{"path":"d:/1.tif","style":{"kind":"trueColor", "bandMap" : [1, 2, 3] , "bandCount" : 3, "stretch" : {"kind":"percentMinimumMaximum", "percent" : 3.0}}},{"path":"d:/2.tif", "style" : {"kind":"trueColor", "bandMap" : [1, 2, 3] , "bandCount" : 3, "stretch" : {"kind":"percentMinimumMaximum", "percent" : 3.0}}}]}

## 前端调用方式二：通过接口注册。使用这种方式并且部署多个服务时，需要指定etcd启动；只启动一个服务，不用启动etcd。Windows下etcd需要使用—enable-v2方式启动（etcd默认是以v3方式启动），目前影像服务linux版支持etcd v3，windows版不支持etcd v3。user和group可以任意指定，根据业务需要来定。AddImages，GetImages，ClearImages 这三个接口只是简单的记录数据的分组，可以不调用这三个接口，可以直接调用UpdateDataStyle，产生一个uid，前端把这个uid写到url里，就不用每次都指定数据和style。

### AddImages

<http://localhost:8083/v1?request=AddImages>

方法：post

body:

{"user":"ln4", "group":"一月", "images":["1.tif", "2.tif", "3.tif", "d:/xxx.tif"]}

### GetImages

<http://localhost:8083/v1?request=GetImages>

方法：post

body:

{"user":"ln4", "group":"一月"}

### ClearImages

<http://localhost:8083/v1?request=ClearImages>

方法：post

body:

{"user":"ln4", "group":"一月"}

### UpdateDataStyle

<http://localhost:8083/v1?request=UpdateDataStyle>

方法：post

body:

{"info":[{"path":"d:/linux\_share/world.tif","style":{"stretch":{"kind":"standardDeviation","scale":0.5}}},{"path":"d:/linux\_share/t/1.tiff","style":{"stretch":{"kind":"percentMinimumMaximum","percent":3.0}}}]}

返回值：B09650397BE37501724001783F22E8C0

### mapbox前端地址

[http://localhost:8083/v1?request=GetTile&TILECOL={x}&TILEROW={y}&TILEMATRIX={z}&key=B09650397BE37501724001783F22E8C0](http://localhost:8083/v1?request=GetTile&TILECOL=%7bx%7d&TILEROW=%7by%7d&TILEMATRIX=%7bz%7d&key=B09650397BE37501724001783F22E8C0)

# WMS调用接口：

将“request=GetTile&TILECOL={x}&TILEROW={y}&TILEMATRIX={z}”改为“service=wms&request=GetMap&BBOX=0.0,0.0,800.0,600.0&width=500&height=1000”。其中BBOX顺序为left,bottom,right,top。坐标默认为webmecator。其他参数与wmts类似

# 影像信息查询接口：

[http://localhost:8083/v1?request=](http://localhost:8083/v1?request=UpdateDataStyle)GetLayInfo

方法：post

body：

["c:/test/world.tif", "c:/test/abc.tif"]

返回值：

[{"left":0.0,"right":500.0,"top":1000.0,"bottom":0.0,"epsg":4326},{"left":0.0,"right":500.0, "top": 1000.0, "bottom": 0.0, "epsg":-1}]

[http://localhost:8083/v1?request=](http://localhost:8083/v1?request=UpdateDataStyle)GetEnvelope

方法：post

body：

["c:/test/world.tif", "c:/test/abc.tif"]

返回值：格式与上面相同，但是单位是经纬度

# 备注：

get和post的json格式是一样的，如下例子1,2,3

例子1：

{"info":[{"path":"d:/1.tif","style":{"kind":"trueColor","bandMap":[1, 2, 3] ,"bandCount":3, "stretch":{"kind":"percentMinimumMaximum","percent":3.0}}},{"path":"d:/2.tif","style":{"kind":"trueColor","bandMap":[1,2,3],"bandCount":3,"stretch":{"kind":"percentMinimumMaximum","percent":3.0}}}]}

例子2：

{"info":[{"path":"d:/1.tif","style":{"stretch":{"kind":"percentMinimumMaximum","percent":3.0}}},{"path":"d:/2.tif","style":{"stretch":{"kind":"percentMinimumMaximum","percent":3.0}}}]}

例子3：

{"info":[{"path":"d:/linux\_share/DEM-Gloable32.tif","style":{"stretch":{"kind":"standardDeviation","scale":0.5}}},{"path":"d:/linux\_share/t/1.tiff","style":{"stretch":{"kind":"percentMinimumMaximum","percent":3.0}}}]}

style字段可以省略，则使用默认显示方式

info的stretch支持下面四种拉伸方式

{"style":{"kind":"trueColor","bandMap":[1,2,3],"bandCount":3,"stretch":{"kind":"percentMinimumMaximum","percent":3.0}}}

{"style":{"kind":"trueColor","bandMap":[1,2,3],"bandCount":3,"stretch":{"kind":"minimumMaximum","minimum":[0.0, 0.0, 0.0],"maximum":[255.0,255.0,255.0]}}}

{"style":{"kind":"trueColor","bandMap":[1,2,3],"bandCount":3,"stretch":{"kind":"histogramEqualize","percent":0.0}}}

{"style":{"kind":"trueColor","bandMap":[1,2,3],"bandCount":3,"stretch":{"kind":"standardDeviation","scale":2.05}}}

可以省略一些字段，例如：

{"style":{"stretch":{"kind":"percentMinimumMaximum","percent":3.0}}}