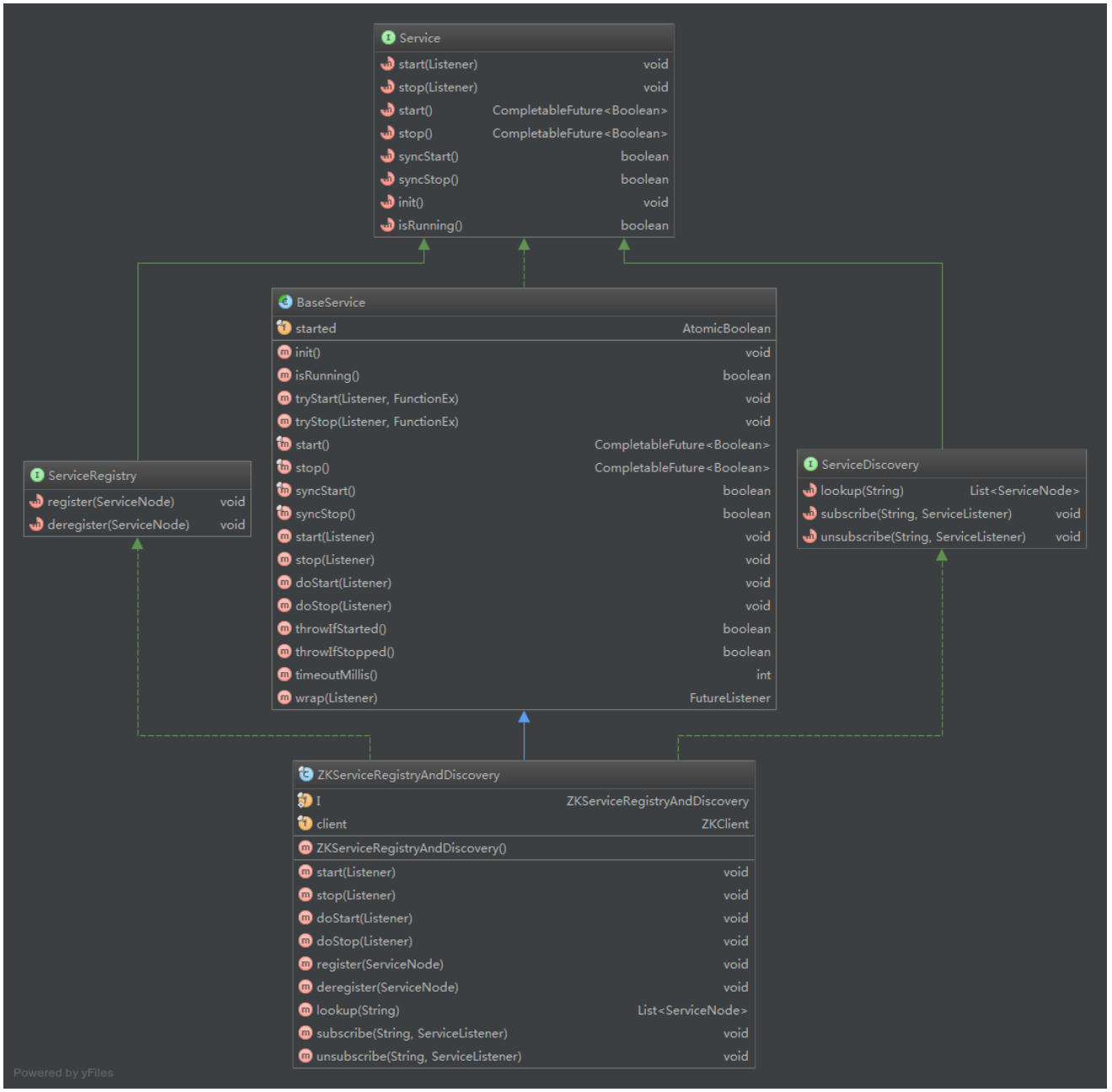
Service
start(Listener) void
stop(Listener) void
start() CompletableFuture<Boolean>
stop() CompletableFuture<Boolean>
syncStart() boolean
syncStop() boolean
init() void
isRunning() boolean
BaseService
started AtomicBoolean
init() void
isRunning() boolean
tryStart(Listener, FunctionEx) void
tryStop(Listener, FunctionEx) void
start() CompletableFuture<Boolean>
stop() CompletableFuture<Boolean>
syncStart() boolean
syncStop() boolean
start(Listener) void
stop(Listener) void
doStart(Listener) void
doStop(Listener) void
throwIfStarted() boolean
throwIfStopped() boolean
timeoutMillis() int
wrap(Listener) FutureListener
ServiceRegistry
register(ServiceNode) void
deregister(ServiceNode) void
ServiceDiscovery
lookup(String) List<ServiceNode>
subscribe(String, ServiceListener) void
unsubscribe(String, ServiceListener) void
ZKServiceRegistryAndDiscovery
I ZKServiceRegistryAndDiscovery
client ZKClient
ZKServiceRegistryAndDiscovery()
start(Listener) void
stop(Listener) void
doStart(Listener) void
doStop(Listener) void
register(ServiceNode) void
deregister(ServiceNode) void
lookup(String) List<ServiceNode>
subscribe(String, ServiceListener) void
unsubscribe(String, ServiceListener) void
Powered by yFiles

初始化

```
PushClient  doStart()

@Override
protected void doStart(Listener listener) throws Throwable {
    if (mPushClient == null) {
        mPushClient = new MPushClient();
    }

    pushRequestBus = mPushClient.getPushRequestBus();
    cachedRemoteRouterManager = mPushClient.getCachedRemoteRouterManager();
    gatewayConnectionFactory = mPushClient.getGatewayConnectionFactory();

    ServiceDiscoveryFactory.create().syncStart();
    CacheManagerFactory.create().init();
    pushRequestBus.syncStart();
    gatewayConnectionFactory.start(listener);
}
```

利用SPI机制，找到ServiceDiscoveryFactory接口实现SimpleDiscoveryFactory或者ZKDiscoveryFactory，得到FileSrd(本地cache.dat)或者ZKServiceRegistryAndDiscovery实例;调用FileSrd或者ZKServiceRegistryAndDiscovery的doStart(Listener listener)方法启动；

```
ZKServiceRegistryAndDiscovery

public static final ZKServiceRegistryAndDiscovery I = new ZKServiceRegistryAndDiscovery();

private final ZKClient client;

public ZKServiceRegistryAndDiscovery() {
    this.client = ZKClient.I;
}

@Override
public void start(Listener listener) {
    if (isRunning()) {
        listener.onSuccess();
    } else {
        super.start(listener);
    }
}

@Override
public void stop(Listener listener) {
    if (isRunning()) {
        super.stop(listener);
    } else {
        listener.onSuccess();
    }
}
                                    截图(Alt + A)
@Override
protected void doStart(Listener listener) throws Throwable {
    client.start(listener);
}
```

调用ZKClient#start(listener)方法

```
ZKClient  start()

@Override
public void start(Listener listener) {
    if (isRunning()) {
        listener.onSuccess();
    } else {
        super.start(listener);
    }
}
```

调用父类BaseService#start(listener)方法，然后调用tryStart()方法

```
BaseService  tryStart()
    }

    protected void tryStart(Listener l, FunctionEx function) {
        FutureListener listener = wrap(l);
        if (started.compareAndSet(false, true)) {
            try {
                init();
                function.apply(listener);
                listener.monitor(this);//主要用于异步，否则应该放置在function.apply(listener)之前
            } catch (Throwable e) {
                listener.onFailure(e);
                throw new ServiceException(e);
            }
        } else {
            if (throwIfStarted()) {
                listener.onFailure(new ServiceException("service already started."));
            } else {
                listener.onSuccess();
            }
        }
    }
```

调用子类init方法

调用子类doStart

init()方法主要是获取ZK配置、初始化ZK实例对象

```
ZKClient  init()
     */
    @Override
    public void init() {
        if (client != null) return;
        if (zkConfig == null) {
            zkConfig = ZKConfig.build();
        }
        Builder builder = CuratorFrameworkFactory
                .builder()
                .connectString(zkConfig.getHosts())
                .retryPolicy(new ExponentialBackoffRetry(zkConfig.getBaseSleepTimeMs(), zkConfig.getMaxRetries(), zkConfig.getMaxSleepMs()))
                .namespace(zkConfig.getNamespace());

        if (zkConfig.getConnectionTimeout() > 0) {
            builder.connectionTimeoutMs(zkConfig.getConnectionTimeout());
        }
        if (zkConfig.getSessionTimeout() > 0) {
            builder.sessionTimeoutMs(zkConfig.getSessionTimeout());
        }

        if (zkConfig.getDigest() != null) {
            /*
```

ZK配置

ZK实例

doStart方法主要是启动ZK Client服务、注册连接状态监听器

```
ZKClient  doStart()
    }

    @Override
    protected void doStart(Listener listener) throws Throwable {
        client.start();
        Logs.RSD.info("init zk client waiting for connected...");
        if (!client.blockUntilConnected(1, TimeUnit.MINUTES)) {
            throw new ZKException("init zk error, config=" + zkConfig);
        }
        initLocalCache(zkConfig.getWatchPath());
        addConnectionStateListener();
        Logs.RSD.info("zk client start success, server lists is:{}", zkConfig.getHosts());
        listener.onSuccess(zkConfig.getHosts());
    }
```

订阅注册中心事件

```
GatewayTCPConnectionFactory  doStart()
    public GatewayTCPConnectionFactory(MPushClient mPushClient) {
        this.mPushClient = mPushClient;
    }

    @Override
    protected void doStart(Listener listener) throws Throwable {
        EventBus.register(this);

        gatewayClient = new GatewayClient(mPushClient);
        gatewayClient.start().join();
        discovery = ServiceDiscoveryFactory.create();
        discovery.subscribe(GATEWAY_SERVER, this);
        discovery.lookup(GATEWAY_SERVER).forEach(this::syncAddConnection);
        listener.onSuccess();
    }
```

1 初始化服务发现实例

根据SPI指定的实现类初始化FileSrd或者ZKServiceRegistryAndDiscovery

2 订阅注册的服务节点信息变化

调用ZKServiceRegistryAndDiscovery#subscribe方法，订阅节点添加、删除、修改事件；

3 查找注册的服务，并针对每个服务节点创建多个conn连接

```
@Override
public void subscribe(String watchPath, ServiceListener listener) {
    client.registerListener(new ZKCacheListener(watchPath, listener));
}

@Override
public void unsubscribe(String path, ServiceListener listener) {

}
```

注册ZK 监听，订阅节点添加、删除、修改事件

查找服务

GATEWAY_SERVER为 /cluster/gs

==缓存写入(本地文件方式)==

D:\...\mpusher\mpush\mpush-test\target\classes\cache.dat


```
{
  "/cluster/cs": {
  "d0b5047f-1705-4395-aa93-017a277f89c1": "{\"attrs\":{\"weight\":1},\"host\":\"172.16.177.132\",\"port\":3000}"
  },
  "/cluster/ws": {
  "8975a6f6-6d04-4929-b33c-802c058b7aa6": "{\"host\":\"172.16.177.132\",\"port\":8008}"
  },
  "/cluster/gs": {
```

```
  "1722a969-e6a9-476a-bd8c-050bacbd54a0": "{\"host\":\"172.16.177.132\",\"port\":3001}"
  }
}
```

==缓存写入(ZK节点方式)==

/cluster/gs # 服务名

---1722a969-e6a9-476a-bd8c-050bacbd54a0 #子节点

"{\"host\":\"172.16.177.132\",\"port\":3001}" #子节点数据

---.....

ZKServiceRegistryAndDiscovery lookup()

```java
@Override
public List<ServiceNode> lookup(String serviceName) {
    List<String> childrenKeys = client.getChildrenKeys(serviceName);
    if (childrenKeys == null || childrenKeys.isEmpty()) {
        return Collections.emptyList();
    }

    return childrenKeys.stream()
            .map(key -> serviceName + PATH_SEPARATOR + key)
            .map(client::get)
            .filter(Objects::nonNull)
            .map(childData -> Jsons.fromJson(childData, CommonServiceNode.class))
            .filter(Objects::nonNull)
            .collect(Collectors.toList());
}
```

获取子节点