将ATT层的attribute构建wield一个个的server和characteristics

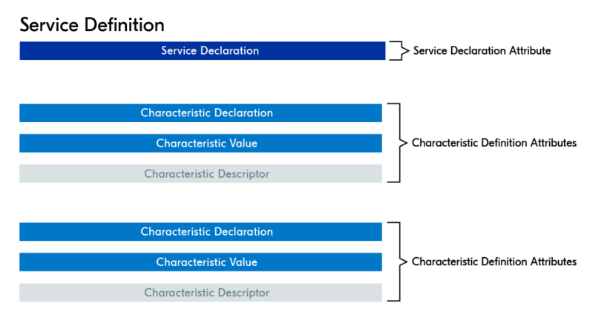
**服务发现是基于GATT层的。**GATT 客户端在 GATT 服务器托管的属性表中发现服务和特征的过程，S和C之间的交互基于attribute进行的

**客户端启动的操作 ：**

* 读（Read）——C向S发起读取请求，S会返回属性值来进行响应
* 写（Write）——C向S发送一个写入请求，并提供匹配的数据，S接受写入就会进行响应确认
* 写入而不响应（Write without response）——C向S写数据，但是不用等待S的确认操作

**服务端启动的操作——虽然是由服务器发起，但是需要客户端订阅或者启用**

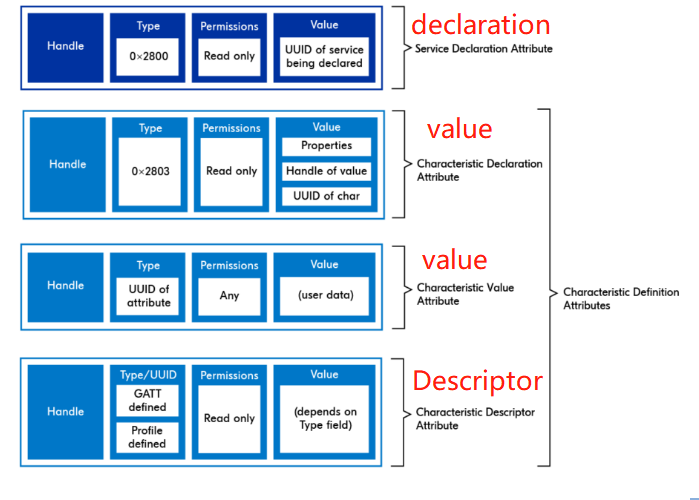
* 通知（Notify）——S使用notify把某个属性（就是数据）推送给C，无需C请求，也无需C确认
* 指示（Indicate）——S使用indicate将某个属性推送给C，需要客户进行确认（导致每一个连接间隔只能发送一包，确认信息在下一个连接间隔发送）



# service declaration attribute（服务的声明属性）——标志着一个serive的开始

* **handle ：16bit,对属性寻址的行号**
* **type：0x2800，**通用唯一 ID （UUID），SIG定义的，它告诉我们属性类型。例如，如果此属性声明了一个特征，这将反映在其**“类型**”字段中，因为它将包含一个专门用于指示声明特征的 UUID，标志服务的开始
* **权限：只读，为什么设置只读权限，因为这只是告诉客户端，接下来是一个什么服务，是服务端定义好后就不能改的，所以只有读权限**
* **value：声明服务的完整UUID**

**characteristics (特征) ——可以有多个attribute，表示服务中的不同数据（可以理解为服务就是一本书的一个章节，服务就是一个章节的头，然后其余操作是这个章节下面的各小结知识点，可以有许多章节，每一个章节可以有N个小的章节点）**

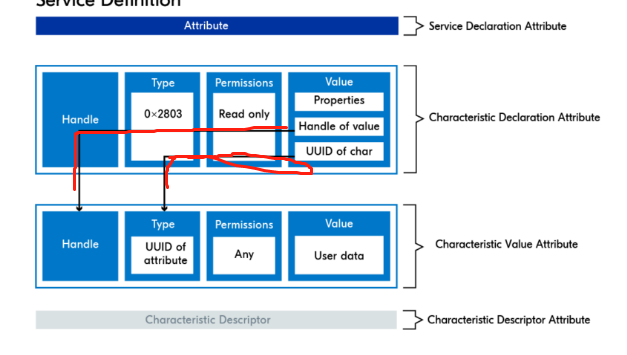


# Characteristic declaration attribute——特征声明属性，表示一个服务中一个数据属性段（ value attribute）的开始

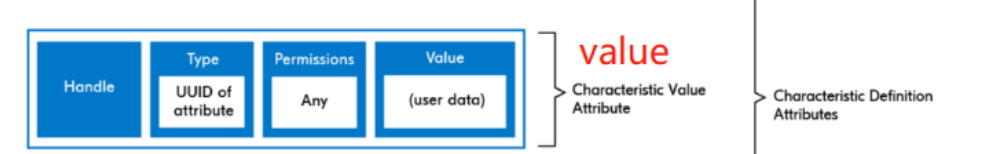
* handle：16bit
* typed：0x2803
* 权限：只读
* value：特征权限——允许对characteristics进行什么操作

特征句柄——句柄地址

特征的UUID——被声明的**characteristics的UUID**



# Characteristic value attribute——特征值属性，用户层给到service的实际数据就在value attribute的value段



* **Handle：操作数据的时候的句柄，自动废品**
* **Type：UUID**
* **Permission:任意权限，用户配置**
* **Value：真正的用户数据部分**

# Characteristic descriptor attribute——特征描述符（可选的选项），就是数据的额外信息，比如温度的单位是什么

CCCD：

一个特殊的特征描述符号CCCD（客户端特征配置描述符），当客户端支持服务端启动的操作 Notify 和 indicate（通知和指示），CCCD就是必须的，客户端可以通过使能和禁止CCCD来订阅他是否希望接受到来着服务端的Notify和indicate。

一个标准的CCCD attribute如下：

句柄：操作地址

type：也就是UUID——0x2902

权限：对CCCD的操作权限，权限是必须为读写

value：表示CCCD是Notification的，还是indicateions的，1为Notification，2为indicateions

