**Session**

**Session概述**

1. **什么是Session：**

-> Session是一种HTTP协议状态的保持机制

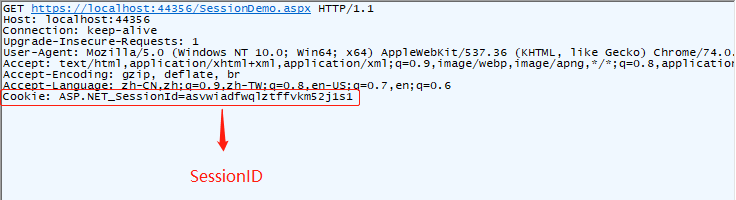
-> Session代表服务器与浏览器的 [会话] 过程，这个过程是连续的，也可以时断时续的

-> Seesion是服务器上，以一段唯一的、不重复的、不容易找到规律的 [SessionID] 对应着一块独立的存储（内存、硬盘、数据库）区域的信息，这个 [SessionID] 是客户端与服务器在通信的过程当中用于区分客户端的信息，每个客户端所建立的 [SessionID] 与其他客户端所建立的 [SessionID] 是互不干扰的、独立的存在

1. **Session的创建过程**

-> 当客户端与服务器进行一次HTTP通信的时候，服务器会根据当前请求的用户创建一个唯一的 [SessionID]，并把所需要存储的信息存放在服务器的内存、硬盘或数据库中，其存储方式是一个 [SessionID] 对应着一块独立的存储区域(以下称为Session池)，在返回响应报文给客户端的时候，服务器会自动新建一个名为 [ASP.NET\_SessionId] 值为刚刚所创建的 [SessionID] 的Cookie信息，并把它放在HTTP响应报文首部字段 [Set-Cookie] 当中一起返回给浏览器，当浏览器接收到响应报文后发现存在需要创建Cookie的指令则会立刻创建一个Cookie(名为 [ASP.NET\_SessionId] 值为 刚刚所创建的 [SessionID]），当浏览器在下一次请求这个服务器的资源的时候，就会自动地把刚刚所创建的Cookie放在里面，当服务器接收到请求且发现存在一个 名为[ASP.NET\_SessionId] 的Cookie的时候，则会根据这个Cookie所对应的值([SessionID])在Session池中找到所存储的内容，这个内容则是我们上上次发送的请求所保存的信息

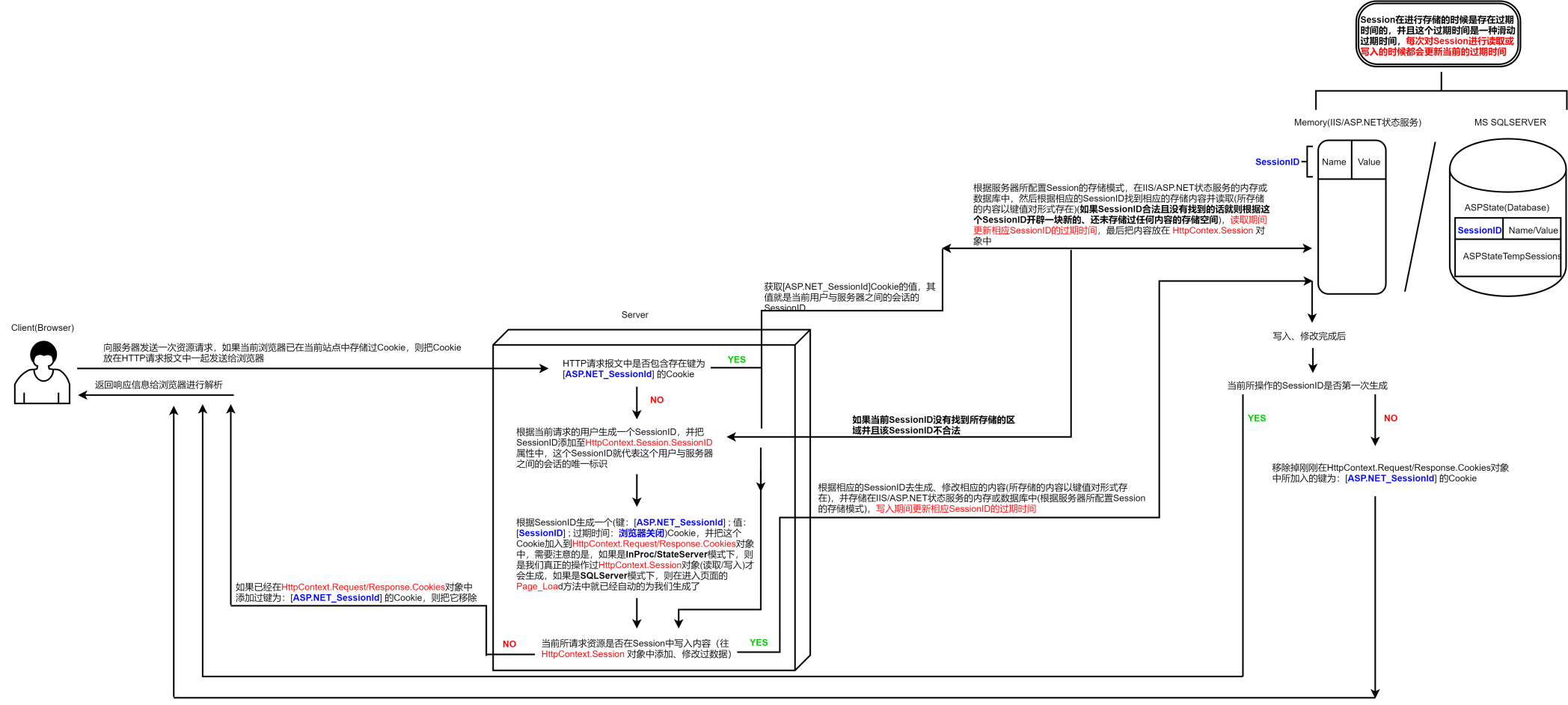
-> Session在请求报文中的体现：



-> Session在响应报文中的体现：



-> Session在ASP.NET中的处理模型

****

1. **Session的存储方式：Session是支持以任何类型进行存储的，并且其在服务器上的存储方式也是可控的**

**->** InProc：存储于当前应用程序域的内存当中

1. 优点

-> 容易实现，默认的存储模式就是这种

-> 这种模式执行得更快，因为它是存储到当前所部署的应用程序域的内存当中

-> 无需对存储的内容进行序列化和反序列化

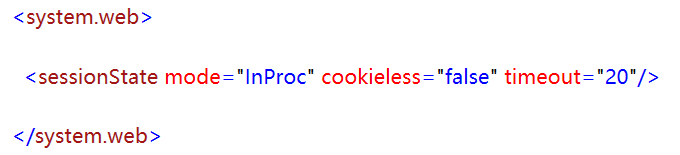
1. 缺点

-> 服务器的重启或应用程序域的回收（更新配置文件、更新页面文件）会导致Session的丢失

-> 会产生较多的会话碎片，增加应用程序域的内存负载

1. 配置方式

-> 默认的存储模式就是这种，无需我们手动的再去设置，但我们也是可以通过 [Web.config] 中来配置的



-> SateServer：存储与ASP.NET状态服务的内存当中

1. 优点

-> 当应用程序域重启的时候也不会造成Session的丢失，但是服务器的重启还是会造成Session的丢失，因为他存储的到的是Windows服务进程的内存当中

-> 支持远程主机服务，也就是说Session的存储不再依赖于应用程序域所在的服务器

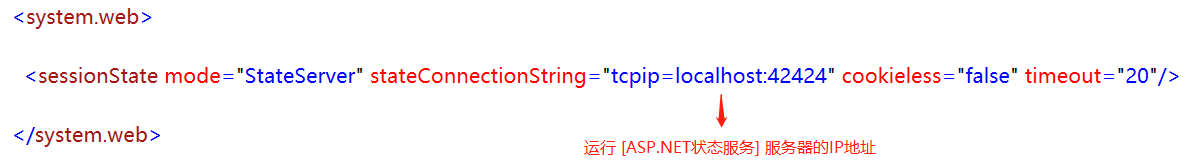
1. 缺点

-> 必须对存储的内容进行序列化和反序列化才能进行存储

-> 两个进程间传递信息时将导致延迟，如果频繁访问和更改状态信息，则速度会明显变慢

1. 配置方式

-> 配置方式通过一个配置文件就可以实现，但是必须配置 [StateConnectionString] 属性的值，需要了解的是 [ASP.NET状态服务] 的默认端口号是42424



-> SqlServer：存储于数据库当中

1. 优点

-> 只要SqlServer是正常运行就不会造成Session丢失的

-> 由于存储在数据库中，具有更加好的弹性

1. 缺点

-> 必须对存储的内容进行序列化和反序列化才能进行存储

-> 属于最慢的一种存储方式

1. 配置方式

-> 打开cmd命令行窗口

-> 输入：cd C:\Windows\Microsoft.NET\Framework64\ [.NET Framework版本]

-> 输入：aspnet\_regsql.exe -S [数据库地址] -U [数据库账号] -P [数据库密码] -ssadd -sstype p

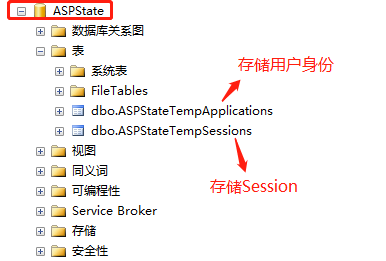
-> 关于更多的命令参数解释，参考下图：



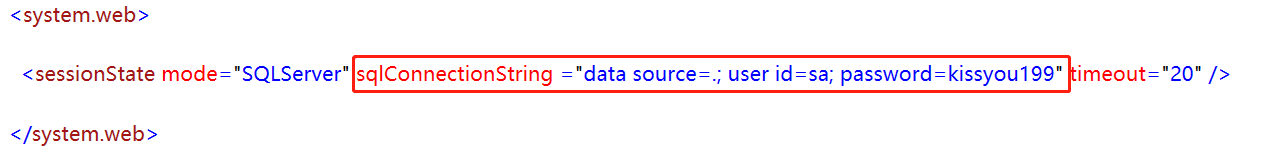
-> 如果没出现错误的话，成功信息如图：



-> 此时我们已经在SqlServer中创建了一个数据库了，其结构如下：



-> 这时我们再对配置文件简单的设置以下数据库的 [连接字符串] 就能使用该模式进行Session的存储了



1. **关于Session的一些问题：**

-> Session 不是免费的，它迫使 Web 服务器在服务器内存中保存额外的信息，即使信息量很小，成百上千个用户访问网站时也很可能会迅速造成一场灾难

-> Session与Cookie是密切关联的，要保证用户浏览器的Cookie功能是开启状态才可以使用Session

1. **在ASP.NET开发中使用Session的一些注意事项**

-> 在一次新的会话中，ASP.NET会为我们生成一个新的SessionID，那么什么才是一次新的会话？

1. 当浏览器发送给服务器的请求信息中不包含 [ASP.NET\_SessionId] 的Cookie
2. 当前 [ASP.NET\_SessionId] Cookie的SessionID并没有在Session池中存储东西（可能是Session过期或是 SessionID是非法的）

-> 使用ASP.NET操作Session的时候，当接收到的请求报文中的字段带有时候[ASP.NET\_SessionId]的Cookie的时候，则会自动的在[HttpContext.Request.Cookies]对象中添加一个键为[ASP.NET\_SessionId]的Cookie，在什么情况下才会自动的在[HttpContext.Response.Cookies]对象中添加一个键为[ASP.NET\_SessionId]的Cookie？

1. InProc/StateServer存储模式

-> 当前会话是第一次会话并且读取过[HttpContext.Session.SessionID]属性的值

-> 当前会话是第一次会话并且请求报文中带有[ASP.NET\_SessionId]的Cookie并且这个Cookie的值是非法的**（特殊情 况）**

1. SQLServer存储模式

-> 只要一加载页面就会自动的在[HttpContex.Request/Response.Cookies]对象中添加SessionID的Cookie

-> 只要在[HttpContex.Response.Cookies]对象中存在一个键为[ASP.NET\_SessionId]的Cookie，返回给浏览器的响应报文中就会带上这个Cookie吗？

1. InProc/StateServer存储模式

-> 当前会话是第一次会话并且写入了一个Session

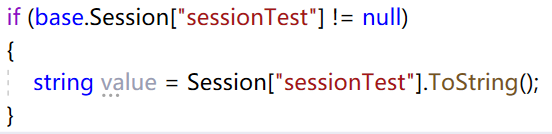
-> 当不是第一次会话的时候，只要在本次操作[HttpContex.Response.Cookies]对象中存在SessionID的Cookie，那么 返回给客户端的响应报文中 则必定会带上 [Set-Cookie] 的首部字段

1. SQLServer存储模式

-> 当前会话是第一次会话并且写入过Session的时候

-> 当不是第一次会话的时候，只要我们修改了[ASP.NET\_SessionId]Cookie的任何属性，都会把[HttpContex.Respons e.Cookies]对象中存储的[ASP.NET\_SessionId]Cookie放在响应报文中返回给浏览器

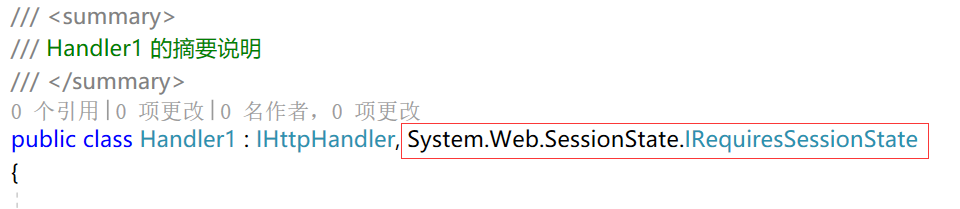
-> 由于设置了Session之后，Session是存在过期时间的，并且Session所存的类型都为object类型，通常需要我们转换成其他类型去操作，但是如果这个Session已经过期了或者说这个Session压根就没存储过指定Name的值，那么我们取出来的值就会是null，这时候转换其他类型就会报错，所以我们在后端的代码里面要获取Session的时候最好先判断下Session是否等于null



-> 在添加一个Session的时候，我们最好不要采用通过【Add】函数来添加的方式，因为这种方式是很有可能存在Session键重复的问题的



-> 如果想在ASP.NET中操作Session，必须显式/隐式的继承自 [System.Web.SessionState.IRequiresSessionState] 接口，否则的话 [HttpContext.Session] 这个属性的值为null，我们就无法操作Session了



1. **网页参考：**

-><https://www.cnblogs.com/panchunting/archive/2012/06/25/ASPNET_Session_Mode.html>

-><https://www.cnblogs.com/SkySoot/archive/2012/07/13/2590117.html>

-><https://blog.csdn.net/zhifeiya/article/details/25965895>

->[https://www.c-sharpcorner.com/UploadFile/225740/introduction-of-session-in-Asp-Net/https://www.c-sharpcorner.com/UploadFile/225740/introduction-of-session-in-Asp-Net/](https://www.c-sharpcorner.com/UploadFile/225740/introduction-of-session-in-Asp-Net/)