ASP.NET缓存技术

缓存是系统或者应用程序将频繁使用的数据保存到内存中，当系统或应用程序再次请求的时候，能够快速地获取数据。缓存技术是提高web应用程序开发人员开发效率最常用的技术。在ASP.NET中有3中常用的缓存技术，即页面输出缓存、页面部分缓存和页面数据缓存。

缓存就是将那些频繁访问而且不需要经常更新的数据存储在内存中，当客户端再一次访问这些数据时，可以避免重复获取满足先前请求的信息。

优点：

1. 支持更为广泛和灵活地可开发特征（ASP.NET 2.0及以上版本包含一些新增的缓存控件和API，如自定义缓存依赖、substitution控件、页面缓存API等）
2. 增强可管理性。
3. 提供更高的性能和可伸缩性。

缺点：

显示的内容可能不是最新、最准确的，为此必须设置合适的缓存策略。又如，缓存增加了系统的复杂性，并使其难于测试和调试，因此建议在没有缓存的情况下开发和测试应用程序，然后在性能阶段启用缓存选项。

1. 页面输出缓存

该机制将整个ASP.NET页面内容保存在服务器内存中。当用户请求该页面时，系统从内存中输出相关的数据，直到缓存数据过期。在这个过程中，缓存内容直接发送给用户，而不必再次经过页面处理生命周期。通常情况下，页面输出缓存对于那些包含不需要经常修改的内容，但需要大量处理才能编译完成的页面特别有用。

1. 页面部分缓存

通常情况下，缓存整个页是不合理的，因为页的某些部分可能在每一次请求时都进行更改，这种情况下，只能缓存一部分即页面部分缓存。

页面部分缓存的实现包括控件缓存和缓存后替补两种方式。前者也可称为片段缓存，这种方式允许将需要缓存的信息包含在一个用户控件内，然后将该用户控件标记为可缓存的，以此来缓存页面输出的部分内容。例如，要开发一个股票交易的网页，每支股票价格是实时变动的，因此，整个页面必须是动态生成且不能缓存的，但其中有一小块用于放置过去一周的趋势图或成交量，它存储的是历史数据，这些数据早已是固定的事实，或者需要很长一段时间后才重新统计变动，将这部分缓存下来有很高的效益，可以不必为相同的内容做重复的计算从而浪费时间，这是就可以用控件缓存。缓存后替补与用户控件缓存正好相反，指的是在控件区域内的数据部缓存而次区域外的数据缓存。

1. 页面数据缓存

页面数据缓存即应用程序数据缓存，它提供了一种编程方式，可通过键/值对将任意数据存储在内存中。使用应用程序缓存与使用应用程序状态类似，但是也应用程序状态不同的是，应用程序数据缓存中的数据是容易丢失的，即数据不是在整个应用程序生命周期中都存储在内存中。应用程序数据缓存的优点是由ASP.NET管理缓存。

ASP.NET中提供了类似于Session的缓存机制，即页面数据缓存。利用数据缓存，可以在内存中存储各种与应用程序相关的对象。对于各个应用程序来说，数据缓存只是在应用程序内共享，并不能在应用程序间进行共享。Cache类用于实现Web应用程序的缓存。