TaskStore design doc: handling dates

2011-10-17

The initial implementation of TaskStore used a Date data type in the SQL Azure database to store the Due field of a task, which is a due date (without a time). (The Task structure also has the Created and LastModified DateTime fields – which are used by the system for optimistic concurrency control / conflict resolution, and do not figure further into this document).

The issue with this implementation choice was timezone handling. The Date datatype in SQL does not store a timezone component – only a Date. When a client wanted to send a new structure or update an existing one, the timezone component would be sent in serialized JSON (“/Date(xxx-TZ)/”, where xxx is the number of seconds since 1970, and TZ is in hours offset from GMT \* 100). The server would then convert this date into the current timezone (on Azure this would be GMT). And only then would the date be stored in SQL Azure (without preserving TZ info). When the date would be extracted back out, it would show up as /Date(xxx+000)/ (because the current timezone is GMT), which would then be properly interpreted by the client and offset accordingly. The systemic issue was that SQL Azure would strip off the timezone so the date would always get mangled – depending on how much the local timezone is off from GMT (Azure).

A solution of preserving the timezone information was considered, but since it involved a data type change on the service and in the database, a simpler solution was selected instead: storing the due date as a string (called DueDate). This created the smallest impact to the existing code, since a property called Due of the right datatype could just be manufactured on the Task entity for the client-side code. This is in fact what happens both in .NET (WP7) as well as javascript (website).