COMPUTERWORLD

Health Monitoring - overset feature

.NET Frameworket er kæmpe stort og man kan let overse ting, nogle ting giver dog for store muligheder til at de bør overses - en af de muligheder er i mine øjne Health Monitoring.

Skrevet den **02. Jul 2009** af **keysersoze** I kategorien **Programmering / ASP.NET** | ★★☆☆☆

I desværre rigtig mange applikationer lægges der meget lidt eller måske endda slet ingen vægt på overvågning - og det gælder både den generelle brug af applikationen, applikationens helbred men i højere grad også fejl i systemet. Og jo - selv du kan overse mulige fejl-kilder så din applikation fejler! Overvågning er da også klart et let område at springe over da det jo som sådan ikke bidrager til noget synligt funktionalitet i applikationen men bare kører usynligt (og måske endda ubrugt) i baggrunden og det har let kunnet koste en masse tid at udvikle eller betydet implementering af 3. parts software - og med det i tankerne er det underligt at så umiddelbart få kender til Health Monitoring i ASP.NET.

Health Monitoring kom med i ASP.NET 2.0 og er, modsat hvad navnet ellers måske kunne antyde, ikke kun et en låst overvågnings-feature men et lille og udvidelsesmuligt framework til at fange og sende events i din applikation - kort fortalt vil det være muligt fx at sende egne events, fange systemfejl og sende "heartbeats" med nyttig runtime information og informationerne kan fx sendes pr email, gemmes i SQL Server eller i Windows Event Log. Opsætningen af Health Monitoring sker alene i din web.config og det er meget simpelt og lynhurtigt - for at have bare et lille overblik over hvad det hele handler om kan vi opsætte et hurtigt eksempel.

Fang alle fejl og send til en email

Først opsættes mail-informationerne:

```
<system.net>
  <mailSettings>
    <smtp deliveryMethod="Network">
      <network defaultCredentials="true" host="smtpservername" />
    </smtp>
  </mailSettings>
</system.net>
```

Herefter selve Health Monitoring:

```
<system.web>
  <healthMonitoring enabled="true">
    <eventMappings>
      <add name="AlleFejl" type="System.Web.Management.WebBaseErrorEvent,
System.Web,Version=2.0.0.0,Culture=neutral, PublicKeyToken=b03f5f7f11d50a3a" startEventCode="0"
endEventCode="2147483647" />
    </eventMappings>

    <providers>
      <add name="MinMailWebEventProvider"
```

```
type="System.Web.Management.SimpleMailWebEventProvider" to="minmail@domain.dk"
from="noreply@domain.dk" buffer="false" subjectPrefix="Applikationsfejl på domain.dk" />
    </providers>

    <rules>
      <add name="Fang alle fejl" eventName="AlleFejl" provider="MinMailWebEventProvider"
minInterval="00:05:00" />
    </rules>
  </healthMonitoring>
</system.web>
```

Med dette i vores web.config vil vi nu modtage en fejlmeddelelse indeholdende vigtige informationer via email om fx selve fejlen og hvorfra forespørgslen kom hver gang der opstår et problem i vores applikation. Vi vil dog maximalt modtage en mail hver femte minut pr opstået event, det vil altså sige, at hvis den samme event opstår 7 gange inden for 5 minutter fra første gang eventen opstod vil vi stadig kun modtage én mail - dette forhindrer os i at blive helt oversvømmet af mulige periodiske fejl.

Kort fortalt om de tre elementer i Health Monitoring, så benyttes eventMappings til at opsætte en brugbart og let læseligt navn for de ønskede events, providers er en liste bestående af hvad vi kan kalde "modtagere" af de ønskede events og i rules-sektionen sammensætter vi hvilke modtagere, der skal modtage hvilke events. Dette betyder, at vi har rigtig mange muligheder for konfigurationen - vi kan fx gemme informationer flere steder samtidig eller dirigere fejlbeskeder hen til relevante modtagere ud fra eksempelvis typen af fejl.

Log fejl i database

Ovenstående eksempel er ret lige til - lidt kode i web.config-filen og vi er kørende - men vil vi i stedet gemme Health Monitor oplysningerne i databasen skal vi forberede databasen på det først. Databasen forberedes ved at køre ASP.NET SQL Server Registration Tool (aspnet_regsql.exe i %WINDOWS%\Microsoft.NET\Framework\version) - dette gøres i en command-promt ved at eksekvere "aspnet_regsql.exe -E -S servernavn -d databasenavn -A w" for en trusted connection eller "aspnet_regsql.exe -U brugernavn -P password -S servernavn -d databasenavn -A w" hvis der benyttes SQL authentication. Denne operation tilføjer fx nødvendige tabeller og stored procedures til databasen.

Til sidst skal vi lige have tilpasset web.config-filen så vi dels sørger for at have en connection til databasen og dels sørge for at Health Monitoring også benytter denne connection.

Først forbindelsen til databasen:

```
<connectionStrings>
  <add name="MinConnection" connectionString="Data Source=localhost;Initial
Catalog=DinDatabase;Integrated Security=SSPI;" />
</connectionStrings>
```

Dernæst sætter vi healthMonitoring-elementet op præcis som i mail-eksemplet, dog med den undtagelse at vi altså nu benytter en anden provider:

```
<system.web>
  <healthMonitoring enabled="true">
    <eventMappings>
      <add name="AlleFejl" type="System.Web.Management.WebBaseErrorEvent,
```

```
System.Web,Version=2.0.0.0,Culture=neutral, PublicKeyToken=b03f5f7f11d50a3a" startEventCode="0"
endEventCode="2147483647" />
    </eventMappings>

    <providers>
      <add name="MinSqlWebEventProvider" type="System.Web.Management.SqlWebEventProvider"
buffer="false" subjectPrefix="Applikationsfejl på domain.dk" />
    </providers>

    <rules>
      <add name="Fang alle fejl" eventName="AlleFejl" provider="MinSqlWebEventProvider"
minInterval="00:05:00" />
    </rules>
  </healthMonitoring>
</system.web>
```

Kun en lille del af det

De to nævnte eksempler er utrolig brugbare og her taler jeg af egen erfaring - selvom jeg selvfølgelig nødigt vil indrømme, at jeg også kan lave fejl - men heldigvis skal det nævnes, at langt de fleste fejl-meldinger jeg støder på er forsøg på misbrug i applikationerne - fx manipulation med URL eller forsøg på spam.

Eksemplerne er ikke kun brugbare i mange specielt mindre applikationer - de er også meget meget grundlæggende og simple eksempler på de muligheder man har i Health Monitoring. Specielt er mulighederne for at lave custom events en god feature - fx kunne man forestille sig relevansen i at sende events afsted når der kom til- og frameldinger på et nyhedsbrev; i stedet for at programmere sig ud det kunne man lave 2 custom events og fange dem med Health Monitoring og hvis man på et tidspunkt kun ville modtage besked om tilmeldinger men ikke frameldinger skulle der kun en ændring i web-config til og ingen kode-ændringer. Også heartbeat-muligheden er spændende da vi med denne kan bestemme et interval hvor ud fra vi vælger at logge om applikationen er kørende - men nu hvor du har fået kendskabet til denne alt for oversete feature skal der også være lidt tilbage til egne opdagelser.

Herfra er tilbage kun at sige held og lykke med at fange dine ukendte fejl.

Nyttige links

http://msdn.microsoft.com/en-us/library/bb398933.aspx - ASP.NET Health Monitoring Overview
http://msdn.microsoft.com/en-us/library/ms998306.aspx - How To: Use Health Monitoring in ASP.NET 2.0