CHAPTER 1

# Exception Handling

FHICT

> De sleutel tot succes ligt in alle momenten waarop je hebt gefaald. (Jean Marie Molina)

`FileNotFoundException`, `NullPointerException`, enzovoort. Al die fouten die jij ziet in de `Visual Studio debugger` ziet de gebruiker ook. Maar als de gebruiker ze ziet dan crasht zijn programma! Als software engineer dien je mogelijke exceptionele situaties te voorkomen. C# heeft hiervoor de keywords `try`, `catch` en `finally`.

Regelmatig wil je de structuur van je code niet helemaal aanpassen aan exceptionele situaties die je (inderdaad) als een uitzondering wilt beschouwen. Een voorbeeld is wanneer je programma een `connectie` met een `database` nodig heeft, of `requests` naar internet stuurt: je wil dan niet elke keer checken of de connectie naar database of internet nog wel werkt, maar [als] de connectie verloren is gegaan dan moet je programma daar wel mee om kunnen gaan. Je kunt dit doen door om je code een `try-catch-clause` te zetten. Het idee ziet er alsvolgt uit:

```
... // code die verbinding maakt
try {
  // code die er van uit gaat dat de connectie werkt:
  ...
  // als er tijdens het uitvoeren van deze code
  // een exception optreedt springt C# meteen
  // naar het vangnet hieronder dat bijvoorbeeld
  // de gebruiker meldt dat de verbinding
  // verbroken is en dat ie het bijvoorbeeld
  // nog eens moet proberen.
```

```
} catch(Exception exc)
{
  ...
  // code die uitgevoerd wordt wanneer er
  // een probleem optreedt!

}
```

## Andere bronnen

Exceptions at CSharp-station[1]

Exceptions at MSDN[2]

[1] http://www.csharp-station.com/Tutorial/CSharp/Lesson15
[2] https://msdn.microsoft.com/en-us/library/ms173162.aspx