# Week 9: Exceptions en herhaling enums, en de ‘is’ en ‘as’ operator

## Leermiddelen

Heb je hulp nodig bij een onderwerp van deze week, dan bieden onderstaande links hulp.  
(klik op de tekst met ctrl+click om de links te openen)

* De sheets
* HFC# hoofdstuk 8 “Enums and collections”, pagina 327 t/m 345, pagina 353 t/m 354.
* Headfirst C# hoofdstuk 10 (!)
* Praktisch UML, paragraaf H4.4.13

Exceptions catching / throwing / definitie  
<http://msdn.microsoft.com/en-us/library/ms173160.aspx>

<http://www.blackwasp.co.uk/CSharpExceptionHandling.aspx>

<http://www.blackwasp.co.uk/CSharpThrowingExceptions.aspx>

De ‘is’ en de ‘as’ operator.  
<http://msdn.microsoft.com/en-us/library/cc488006.aspx>

<http://msdn.microsoft.com/en-us/library/scekt9xw(v=vs.110).aspx>

<http://msdn.microsoft.com/en-us/library/cscsdfbt(v=vs.110).aspx>

Enums  
<http://msdn.microsoft.com/en-us/library/sbbt4032.aspx>

## Opgave 1

In deze opgave ga je oefenen met de ‘is’ en de ‘as’ operator.

Je kunt in deze oefening de eerder gemaakte klassen Animal, Cat, Dog en SimpleDate hergebruiken.



Maak een nieuw project en zet daarin de bovenstaande klassen. Vul de lijst van dieren in de TestForm met een aantal honden en katten, zodat de applicatie bij het opstarten al dieren in de lijst bevat.

Maak vervolgens een knop op de TestForm. Indien er op deze knop gedrukt wordt moet het programma de lijst van dieren doorlopen met één foreach lus. In de foreach lus moet je van elke hond de waarde van de property LastWalkDate tonen en van elke kat moet de waarde van de property BadHabits getoond worden. Gebruik de ‘is’ en de ‘as’ operator.

## Opgave 2

Kopieer de code van opgave 1.

Van dieren wil men vastleggen wat het geslacht is. De klasse Animal moet dus een extra property *Gender* krijgen.

Omdat het opslaan van de string “Male” of “Female” voor elk dier ‘duur’ is, krijgt de property *Gender* niet *string* als type, maar een enumeratie Gender als type. Zie onderstaand klasse diagram.

Implementeer het bovenstaande klassediagram. Pas, waar nodig, constructors en ToString() methoden aan.

## Opgave 3

Zoek van de volgende C#/.Net constructors en methoden uit welke excepties ze kunnen opgooien.

Gebruik hierbij de online MSDN C# help.

* int Convert.ToInt32(string value)
* string String.Substring(int startIndex, int length)

Maak een programma dat deze excepties veroorzaakt, vang deze gericht af en toon de informatie van de afgevangen exceptie.

## Opgave 4

Pak het bestand “PRC2 – W9 - BigBucksBankWithoutExceptions.zip” uit en open de solution.

Voeg, waar nodig, het opvangen van exceptions toe aan deze code.

En voeg, waar nodig, het opgooien van excepties toe aan deze code.

Maak zelf een exceptions met de naam *NotEnoughCreditException* en laat de *Withdraw*

methode deze exceptions opgooien indien er niet genoeg tegoed aanwezig is voor de opname. De *NotEnoughCreditException* moet een constructor hebben waarin de programmeur een string mee kan geven (voor de *message* van de exception).

Maak nu Unit Tests voor de Account klasse. Test ook met de Unit Tests of de excepties opgegooid worden wanneer dit verwacht wordt.