# Inleiding

In dit document vind je één of meerdere challenges die passen bij het onderwerp **File IO**. Je kunt de challenges gebruiken zoals gegeven, of gebruiken als voorbeeld en inspiratie voor challenges die je zelf opstelt.

## Challenge 1

## 

Neem als uitgangspunt de challenge van het dierenasiel. De applicatie moet voorzien worden van drie extra features:

* De gebruiker moet alle dieren in de administratie op kunnen slaan in een file.
* De gebruiker moet alle dieren in de administratie kunnen vervangen door de dieren die op zijn geslagen in een file.
* In de code moeten exceptions op de juiste manier toegepast worden. Zowel in de nieuwe code als in de bestaande dierenasiel code.

Voeg daarvoor de volgende methoden toe aan de Administration klasse:

/// <summary>

/// Saves all animals to a file with the given file name using serialisation.

/// </summary>

/// <param name="fileName">The file to write to.</param>

public void Save(string fileName)

{

}

/// <summary>

/// Loads all animals from a file with the given file name using deserialisation.

/// All animals currently in the administration are removed.

/// </summary>

/// <param name="fileName">The file to read from.</param>

public void Load(string fileName)

{

}

Implementeer deze methoden en gebruik serialisatie voor het wegschrijven en laden van de gegevens. Zorg ook voor correcte exception afhandeling.

Men wil de gegevens uit de administratie ook exporteren naar andere programma’s, die niet het binaire formaat van de geserialiseerde C# files kennen. Een manier om dit te doen is een tekst file weg te schrijven. Ieder ander programma (ook niet C# programma’s) kan immers een tekst file uitlezen.

/// <summary>

/// Exports the info of all animals to a text file with the given file name.

///

/// Each line of the file contains the info about exactly one animal.

/// Each line starts with the type of animal and a colon (e.g. 'Cat:' or 'Dog:')

/// followed by the properties of the animal seperated by comma's.

/// </summary>

/// <param name="fileName">The text file to write to.</param>

public void Export(string fileName)

Implementeer deze methode. Zorg voor correcte exception afhandeling.

Maak vervolgens een nieuw Windows Forms programma “AnimalFileImporter” aan, dat deze file weer kan importeren (inlezen) en gegevens uit de eerder geëxporteerde file aan de gebruiker kan laten zien.

Het importeerprogramma moet:

* De gebruiker vragen om een naam van het te importeren bestand.
* Na het inlezen van het opgegeven bestand
  + Een listbox tonen met daarin de volgende info van alle ingelezen katten: chipnummer, naam, geboortedatum en de slechte gewoonten van de kat.
  + Een listbox tonen met daarin de volgende info van alle ingelezen honden: chipnummer, naam en geboortedatum.

Verder moeten alle bugs die aanwezig zijn in de eerder uitwerking verholpen worden en alle eerder verkregen feedback moet gebruikt zijn om het programma te verbeteren. Deze Oefening is dus ook een reparatiekans voor eerder nog niet aangetoonde leerdoelen en eerder gemaakte fouten.

De criteria voor de beoordeling zijn:

* De gevraagde functionele eisen zijn geïmplementeerd.
* Je hebt excepties op de juiste manier toegepast.
* Je kan het ‘hoe en waarom’ van de gemaakte code verdedigen.
* Het programma vertoont geen bugs (blijft hangen, crashed, …)
* De gemaakte code is netjes: goede encapsulatie (private/public), duidelijke variabele namen, netjes inspringen, geen dubbele code, zelf toegevoegde methoden hebben een duidelijk doel en een duidelijke naam.
* Juist gebruik van loops (for/while/foreach)
* Juist gebruik van keuze structuren (if/else/else if/switch).