Inlämning 1: Fönsterbaserad applikation i Windows

Syfte

Inlämningsuppgiften behandlar programmering av en fönsterbaserad applikation i Windows. Utvecklingen sker i C# med hjälp av Visual Studio 2019 (eller 2022).

Kursmål

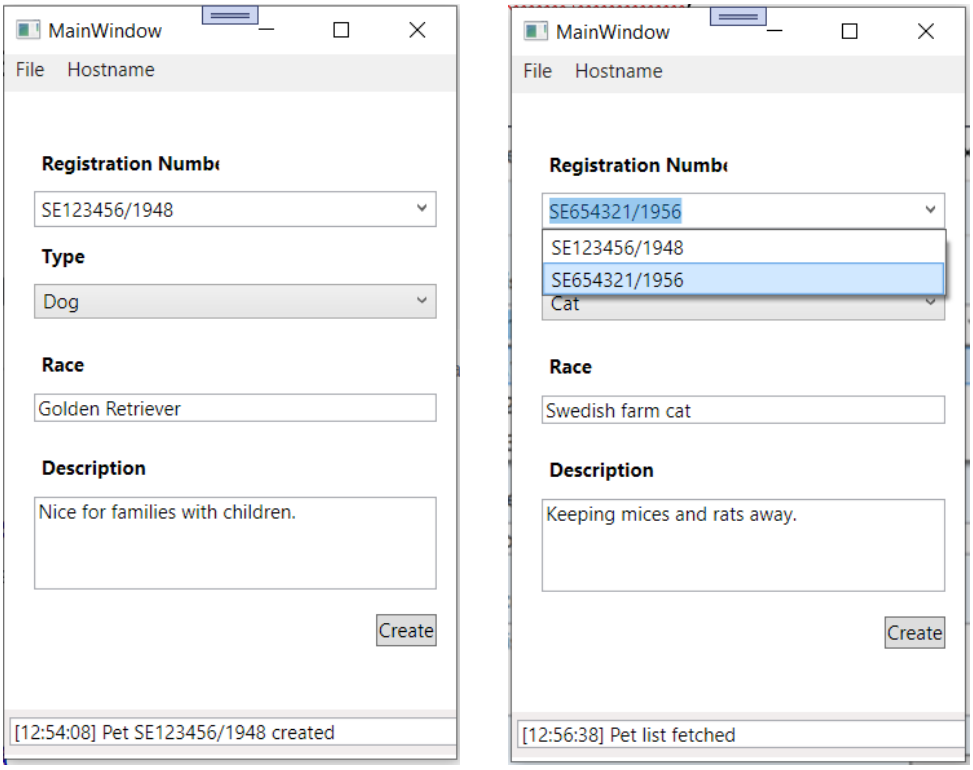
Följande kursmål berörs huvudsakligen i denna inlämningsuppgift:

- 2.1 tillämpa grundläggande konstruktioner i C# och dess standardbibliotek,
- 2.2 tillämpa samlingsklasser samt konstruktioner för in- och utmatning i C# och dess standardbibliotek,
- 2.3 tillämpa konstruktioner för undantagshantering i C# och dess standardbibliotek,
- 2.4 tillämpa konstruktioner för händelsehantering i C# och dess standardbibliotek,
- 2.5 tillämpa konstruktioner för grafisk gränssnittskonstruktion i C# och dess standardbibliotek genom delsystemet WPF
- 2.7 konstruera objektorienterade datorprogram enligt etablerade objektorienterade principer
- 3.1 visa förmåga att bedöma lämpligheten av en objektorienterad lösning utifrån ett givet problem

Uppgift

Ni skall skapa en applikation med hjälp av Windows Presentation Foundation (WPF) som liknar (behöver inte vara exakt lika som) det grafiska användargränssnittet i de två figurerna nedan. Programmet tillåter användaren att mata in uppgifter för husdjur (*Pets*). Varje husdjur (*Pet*) har ett registreringsnummer (*Registration Number*), ras (*Race*) och en tillhörande beskrivning av husdjuret (*Description*). Dessutom kan ett husdjur antingen vara en hund (*Dog*) eller katt (*Cat*). Ras (*Race*) och beskrivning (*Description*) skrivs in i klartext i GUI:t medan typ (*Type*) väljs från en combobox som innehåller två värden (*Dog* och *Cat*). Registreringsnumret skrivs också in i klartext i en redigeringsbar combobox, Dessutom innehåller denna comboboxen en lista med husdjur som användaren har matat in tidigare. Användaren sparar ett nytt husdjur genom att klicka på ”*Create*”-knappen. Längst upp finns en *File* meny. Längst ner finns ett statusfält, där olika meddelanden visas när informationen om husdjuren hanteras datamässigt på olika sätt.

Figuren nedan till höger, visar att den översta comboboxen (*Registration Number*) innehåller en lista med husdjur som användaren har matat in tidigare.



Kravspecifikation

Nedan följer ett antal detaljerade krav på applikationens design och funktionalitet.

- **Meny**. Längst upp skall det finnas en meny som innehåller menyalternativen ”File -> Save” och ”File -> Exit”.

  När man väljer menyalternativet *Save* skall applikationen spara ned all den inmatade informationen i en textfil. Textfilen skall vara formaterad som en komma-separerad lista med filändelsen ”.csv”. Filnamnet på textfilen får användaren automatiskt välja i en standardmässig dialogruta som tillhandahålls av Windows (d.v.s. ingenting som ni själva designar) via .NET.

  När man väljer menyalternativet *Exit* skall applikationen avslutas.

- **Kontroller**. Kontrollerna består av *labels*, *textboxes*, *comboboxes* och en *button* (enligt uppgiftsbeskrivningen ovan). Comboboxen för registreringsnumret måste vara redigeringsbar så att nya husdjur kan läggas till. Dessutom skall denna comboboxen innehålla en lista med alla husdjur som användaren har matat in. Comboboxen för husdjurstypen (*Type*) skall innehålla två värden (*Dog* och *Cat*). Övriga input-fält (*Race* och *Description*) accepterar input i klartext. Statusfältet längst ner skall ej vara redigeringsbart, utan används endast för att visa statusmeddelanden till användaren. Knapparna beskrivs nedan.

- **Knappar och händelser**. När användaren klickar på ”Create”-knappen skall ett nytt husdjur skapas (från de ifyllda uppgifterna i GUI:t) och sparas i en lämplig datastruktur i minnet.

- **Objektorientering**. En lämplig objektorienterad struktur skall skapas för husdjuren, samt tillhörande samlingsklass som innehåller alla husdjur som användaren har skapat. Arv skall tillämpas. Dessutom skall goda objektorienterade principer i övrigt följas för all kod. Alla modell-klasser (d.v.s. husdjuren med tillhörande objektorienterade design) skall läggas i ett separat VS-projekt som WPF-applikationen refererar till.

- **Databindning**. När den underliggande datastrukturen (som innehåller alla husdjur) ändras, skall databindning användas för att uppdatera comboboxen för registreringsnumret. Dessutom skall övriga kontroller databindas till comboboxen (för registreringsnumret), så att deras värden uppdateras med korrekta värden då användaren bläddrar mellan elementen i comboboxen (för registreringsnumret).

- **Statusmeddelanden**. Varje gång användaren skapar ett husdjur, skall statusen för operationen (d.v.s. att operationen lyckades) visas i statusfältet längst ner i GUI:t.

- **Felhantering**. Eventuella undantag (*exceptions*) som kan uppstå i koden vid t.ex. funktionsanrop till ramverket .NET vid filhantering med mera, skall hanteras så att inte applikationen kraschar.

Tips

En *ObservableCollection* kan vara användbar i denna inlämningsuppgift.

Om “code-behind“-filen ej känner igen “x:Name” från XAML-filen, starta om VS.

Arbetsform

Ni skall arbeta i grupper om 1-2 studenter per grupp. Anmäl era önskemål om grupptillhörighet via instruktioner på Canvas, alternativt kontakta kursansvarig.

Samarbete

Det är inte tillåtet att dela lösningar eller delar av lösningar mellan grupperna.

Handledning

Handledningstillfällen finns schemalagda i KronoX och bokas gruppvis via kalendern i Canvas. Handledningen utförs på distans via Zoom av olika lärare och assistenter.

Redovisning

Inlämningsuppgiften redovisas gruppvist genom inlämning i Canvas av en fullständig solution-mapp (för Visual Studio) med all källkod, samt körexempel (förslagsvis via videoinspelning i mp4 eller ett antal illustrerande skärm-dumpar).

Bedömning och om-examination

Arbetet bedöms som en helhet. För godkänt betyg skall samtliga krav vara uppfyllda på ett huvudsakligen tillfredsställande sätt. Arbeten som underkänns vid första examinationstillfället kan kompletteras och lämnas in igen vid nästkommande examinationstillfällen (se datum i kursens schema i KronoX och kalendern i Canvas) inom årets kurstillfälle.