**Prof. Dr.-Ing. Sascha Seifert**

**Fakultät für Technik**

Verteile Systeme mit C# und .NET

## Programmbeilage

Die Prüfung (PLP) besteht aus dem Programmieren eines C#/.NET Projektes. Bewertungskriterien finden Sie im Exam.ppt Dokument.

**Name, Vorname: Albrecht, Pierre  
Matrikelnummer: 311296  
Programmname: ObjectDetectionApp**

## Abstrakt

Anwendungszweck („was macht Ihre Anwendung?“):

Die Anwendung dient zur Veranschaulichung des YOLO Objektdetektions-Algorithmus. Alltagsgegenstände können mit beliebigen Android-Smartphones in Echtzeit erkannt werden. Die Vorhersagen (engl. *predictions*) werden dem Nutzer mithilfe von farbigen Rahmen auf dem Display angezeigt. Die Kamera-Auflösung ist frei einstellbar und wirkt sich entsprechend auf die Performance und Genauigkeit des Systems aus.

Hinweis: Diese Applikation ist nur voll funktionsfähig, wenn sich der Client im Netzwerk der Hochschule Pforzheim befindet, und die Darknet-Anwendung auf dem Leistungsrechner im Medizintechnik-Labor gestartet wurde.

## Architektur

Bitte geben Sie ein Architekturmodell als UML-Klassendiagramm an. Zusätzliche UML-Diagramme können Sie zur Erläuterung ggfls. hinzufügen.

**Siehe *classdiagram.png***

## Technologie

Xamarin.Forms für Cross-Platform-Development + verschiedene Plugins:

* Xam.Plugin.SimpleAudioPlayer: Abspielen von Sounds
* Newtonsoft.Json: Einfache Konvertierung von JSON-Files in .NET-Klassen
* Lottie.Forms: Einfügen von Animationen im JSON-Format (z.B. von <https://lottiefiles.com/> )
* DeviceOrientation: Einrasten der Smartphone-Ausrichtung während der Detektion
* FastAndroidCamera: Schneller Zugriff auf den Hardware-Buffer der Kamera, ohne GC overhead

<https://github.com/jamesathey/FastAndroidCamera>

## Sprach- und Frameworkkonstrukte

Die Tabelle soll als Erinnerungshilfe dienen. Damit können Sie überprüfen, wieviel C# Konstrukte Sie einsetzen. Es ist darauf zu achten wo möglich und angebracht C#/.Net Konstrukte einzusetzen. Ich möchte erreichen, dass nicht nur konventionelle Konstrukte wie klassische Schleifen eingesetzt werden, stattdessen auch einmal ein LINQ-Ausdruck / Prädikat genutzt wird. Foreach Schleifen auch z.B. für eigene Typen genutzt werden. Indexer um einfacher auf Unterstukturen zugreifen zu können. Statt Timer kann man Threads einsetzen. Es gibt auch Klassen in .NET wie DoubleAnimation, wenn es nur um eine Animation auf der Oberfläche geht. Zero-Wait bedeutet z.B. auch Einsatz von Asynchroner Programmierparadigmen.

|  |  |
| --- | --- |
| **Konzept** | **Verwendet?** |
| Nullable Typen | - |
| Vererbung | Ja (siehe Klassendiagramm) |
| …. virtual /override /new | DarknetSettings  Android.PreviewPageCustom.cs |
| …. Constructor base | ServiceClient.cs |
| Überladung | ServiceClient.cs |
| …. Constructor chaining | Ja, z.B. ServiceClient.cs |
| Properties mit backing field | DarknetService.cs |
| Properties mit Zugriffsschutz / readonly | ParticleManager.cs |
| Auto Properties | DarknetService.cs |
| Elvis operator | Ja (z.B. bei Event Invokes) |
| Lambda-Ausdrücke | Menu.xaml.cs |
| Delegates / Events | DarknetService.cs |
| Anonyme Methoden / Typen | Menu.xaml.cs |
| Interface-Verwendung (z.B. IEnumerable) | DarknetSettings.cs  IEnumerable, IEquatable, IDisposable |
| Generics (z.B. in Collections) | CrossPlatformManager.cs |
| …. mit Contraints | CrossPlatformManager.cs |
| Indexer | - |
| Custom Iterator | DarknetSettings.cs |
| Exceptions | CrossPlatformManager.cs |
| Reflection | DarknetSettings.cs  SettingsPage.cs |
| XML-Kommentare | ServiceClient.cs |
| Discard feature (C# 7.0) | Menu.xaml.cs  (MenuButton Constructor) |
| Pattern matching (C# 7.0) | SettingsPage.cs |
| Lock / Threading and thread safe programming | ParticleManager.cs  DarknetService.cs |
| Custom attributes | SettingAttribute.cs |

## Sonstige Erklärungen

Können Sie noch weitere Punkte aufführen, die zum Verständnis Ihres Programms oder die für die Benotung beachtenswert sind?

Für die Darknet-Einstellungen (DarknetSettings.cs) wurden eigene Attribute (siehe SettingAttribute.cs) eingeführt, die das nachträgliche Hinzufügen von Properties vereinfachen sollen. Mithilfe dieser Attribute werden auf der Einstellungsseite (SettingsPage.cs) der App automatisch User Controls zum Einstellen von Properties über die Benutzeroberfläche erzeugt.

**Beispiel**: Wird in den DarknetSettings eine neue Integer-Property „MaxNumberOfPredictionsPerFrame“ hinzugefügt, so wird ein zugehöriger Slider in der App generiert, über den der Nutzer die maximale Anzahl an Objektvorhersagen festlegen kann. Ebenso werden aus Enumerationen Picker und aus Strings Text-Einträge generiert.

Um einen möglichst großen Anteil der Applikation in der Portablen Klassenbibliothek (PCL) ausführen zu können und Mehrfachimplementierungen so gut es geht zu vermeiden, müssen die plattformabhängigen Kamerazugriffe und -darstellungen eine gemeinsame Schnittstelle besitzen: *ICameraPreview*. Diese Schnittstelle wird von allen drei plattformabhängigen Projekten realisiert. Damit nun der PCL ein Zugriff auf die konkreten Instanzen der Zielplattformen ermöglicht wird, die diese Schnittstelle implementieren, wurde eine statische, generische Helfer-Klasse *CrossPlatformHelper<T>* eingeführt. Die Zielplattform referenziert zu Beginn der Applikation seine Klasseninstanz im CrossPlatformHelper, sodass diese Instanz ab diesem Zeitpunkt von der PCL über die definierten Schnittstellen verwendet werden kann.