ToDo Liste mit CSV persisitieren

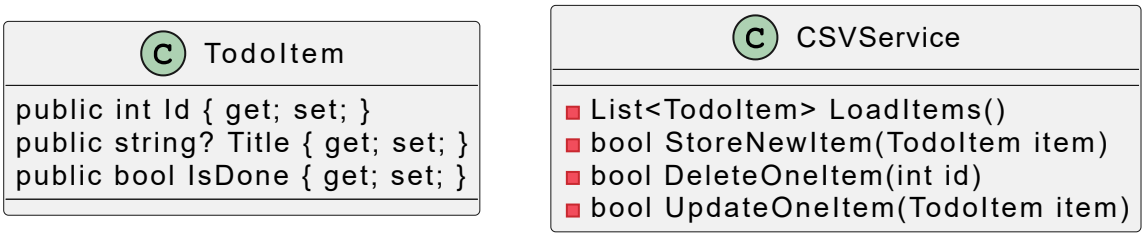
Unter "persistieren" versteht man das dauerhafte Speichern von Daten. Aktuell ist es so dass wenn man die Seite mit der ToDo Liste neu lädt alle Daten verloren gegangen sind. Nicht sehr Sinnvoll wenn man eigentlich später daran erinnert werden möchte. Werden die Daten länger gespeichert kommen wir dem Funktionsumfang einer normalen Webseite immer näher.

Um das zu ändern kann man eine eigene Klasse schreiben welche einen Service startet der sich um das lesen und schreiben der Daten auf die Festplatte (in Dateien) kümmert.

Im Beispielcode ist bereits ein Service enthalten. Der `WeatherForecastService`. Dieser generiert am Server zufällige Daten und sendet diese dann zum Client. Auch die bisherige ToDo-App generierte (wenn auch nur temporär) eine Liste am Server.

Schritt 1: Service erstellen

Lege im Ordner `/Data` eine neue Klasse an mit dem Namen `CSVService`. Diese Klasse wird Daten laden, speichern, ändern und löschen können. Das UML Diagramm dafür könnte z.B so aussehen:



Lesen einer CSV-Datei kannst du von einer bestehenden Abgabe ableiten hier. Zum schreiben von CSV-Dateien gibt es auf StackOverflow schöne Einträge. Um Daten aus der CSV Datei zu ändern oder zu löschen kannst du die komplette CSV-Datei in eine Liste lesen die Änderung bei der entsprechenden Id anwenden und die neue Liste wieder in die CSV-Datei speichern.

Schritt 2: Service zugägnlich machen

Um den fertig programmierten Service verwenden zu können musst du in der Datei `Program.cs` ein Singelton hinzufügen mit `builder.Services.AddSingleton<servicename>();`. In deinem Fall lautet der Servicename wahrscheinlich CSVService. Wenn du die Klasse anders genannt hast musst du das entsprechend anpassen.

Schritt 3: Service in Razorpages aufrufen

In der `ToDo.razor` um den Singleton verwenden zu können musst du den Service injecten. Das geht mit `@inject servicename variablenname`. Im Code-Behind `@code{...}`. Kannst du dann mit z.B `variablenname.LoadItems()` auf den Service zugreifen. Die von `LoadItems()` zurückgegebene Liste kan dann verwendet werden um sie in einer foreach-Schleife wie in der vorherigen abgabe anzuzeigen.